VIC 20

PER GIOCARE CON UN VERO COMPUTER

PER GIOCARE CON UN VERO COMPUTER

Una guida facile al computer

COMMODORE INTERNATIONAL
AVALANCHE, INC.

PREFAZIONE

State per incontrare un computer veramente amico! Amico nel prezzo, nelle dimensioni, amico nell'uso, facile da apprendere e da sperimentare. Ma quello che è più importante - non dovete essere un programmatore di computer e non è necessario che sappiate scrivere a macchina per usarlo!

Se affrontate per la prima volta un computer, questo manuale vi fornirà un'eccellente introduzione alla programmazione. A differenza della maggior parte dei manuali di istruzione, non dovete leggere l'intero libro per arrivare all'argomento che vi interessa. Dopo aver letto il Capitolo 1 (PER COMINCIARE), potete passare direttamente al capitolo che vi interessa ed iniziarne la lettura. Se vi divertite con gli effetti di animazione, passate al Capitolo 4. Se amate la musica, provate il Capitolo 5.

La prima pagina di ciascun capitolo ha un programma campione sul quale potete incominciare a lavorare. Basta che digitiate i programmi esattamente come indicato («Provare a battere questo programma») per vedere cosa succede. Il resto del capitolo spiega ciò che avete fatto e mostra come procedere ulteriormente. Il Capitolo 7 riassume alcuni importanti concetti di programmazione e spiega le tecniche usate nei programmi esemplificativi.

Se avete esperienza di programmazione potete usare il VIC come un qualsiasi microcomputer. La familiarità con i computer Commodore vi sarà di aiuto dato che il BASIC ed i concetti relativi ai grafici sono pressochè identici a quelli usati nel PET/CBM. Nell'Appendice figurano materiali di riferimento e informazioni di programmazione avanzati. Per la programmazione più sofisticata, vi rimandiamo alla GUIDA DI RIFERIMENTO PER IL PROGRAMMATORE VIC, disponibile presso i rivenditori Commodore.

Se non siete un esperto di computer e non avete alcun interesse alla programmazione, dovete dare uno sguardo alla crescente libreria di nastri programma e di cartucce a innesto del VIC. Le cartucce VIC si inseriscono direttamente nella parte posteriore della console e funzionano automaticamente. I programmi sono forniti anche su cassette di nastro magnetico da usare con il registratore a cassetta Commodore.

Cartucce e nastri comprendono giochi interessanti tipo «VIC INVADERS» e programmi educativi per aiutarvi a sviluppare capacità speciali nonché programmi di utilità per la casa che vi aiuteranno a risolvere i problemi più svariati e ad eseguire calcoli.

Le periferiche e gli accessori per il VIC comprendono il registratore a cassetta VIC, l'unità a disco singolo, il modem telefonico e la stampante, per citarne solo i più importanti (vedere Appendice A).

I computer stanno diventando una parte sempre più importante della nostra vita quotidiana – nelle nostre case, a scuola e nella vita. Coloro che hanno familiarità con il computer si troveranno avvantaggiati nei mesi e negli anni a venire. Il VIC non solo vi introduce al mondo della programmazione ma vi dà anche le possibilità e la flessibilità che vi servono per espandere quel mondo.

Approfittatene!

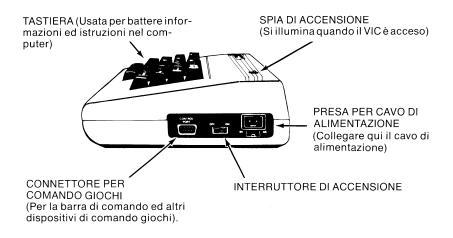
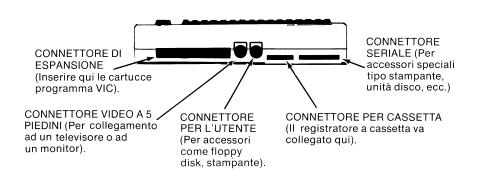
Indice

Capitolo	Titolo	Pagina

Titolo	Pagina

Come collegare il VIC

Benvenuti al calcolo! Le seguenti istruzioni dettagliate mostrano come aprire l'imballo del VIC, collegare il televisore ed assicurarsi che funzioni correttamente.

Iniziamo con il dare un rapido sguardo al VIC-20:

1. Controllare il contenuto della scatola che contiene il VIC. Si dovrebbe trovare quanto segue:
- Personal Computer VIC 20
- Alimentatore (grande scatola da cui escono 2 fili)
- Modulatore RF (piccola scatoletta metallica) e cavo
- Cavo coassiale con spine

2. Occorrono *due prese di corrente* – una per il VIC ed una per il televisore.

3. Posizionare il *VIC ed il televisore* in modo da poter usare la tastiera comodamente mentre si osserva lo schermo del televisore . . . idealmente su una scrivania o un tavolo.

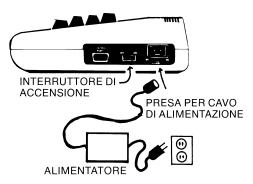
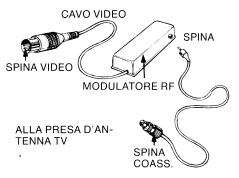
4. Trovare l'interruttore ON/OFF sulla destra del VIC. Assicurarsi che sia nella posizione OFF (spento).

5. Ci sono due cavi che escono dall'alimentatore. Inserire il cavo dell'alimentatore in una presa di corrente e collegare l'altra estremità del cavo nella presa per cavo di alimentazione sul lato del VIC. NOTA: L'alimentatore rimane acceso mentre è inserita la spina cosicchè occorre scollegare quando non è in uso

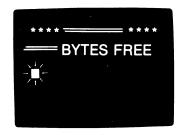
6. Collegare il cavo video alla parte posteriore del VIC ed alla scatola del modulatore RF come indicato. Assicurarsi di collegarlo al connettore video e non al connettore seriale a 6 piedini posto di fianco. PHONO

7. Collegare il modulatore TV al televisore per mezzo della spina fono – la spina si inserisce nel modulatore e l'altra estremità si collega alla presa UHF del televisore per mezzo di una spina coassiale.

8. Accendere il televisore

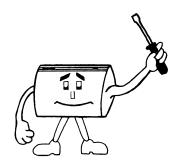
9. Accendere il video (la spia rossa sulla parte superiore del computer dovrebbe accendersi). Se la spia non si illumina, consultare la tabella che segue.

10. Mettere a punto il televisore fino a che non si ottiene un'immagine nitida. La sintonia fine del televisore può richiedere alcune regolazioni. Ecco cosa si dovrebbe vedere comparire sullo schermo – talvolta occorre un secondo o due perchè compaia. Se non si ottiene la seguente visualizzazione sullo schermo, spegnere il computer, attendere alcuni secondi e riaccenderlo.

11. La regolazione del colore e del contrasto dipendono dai comandi previsti sul televisore – naturalmente gli apparecchi con migliori comandi danno una resa cromatica migliore. Alcuni apparecchi rendono taluni colori meglio degli altri.

12. Se ci sono problemi con una qualsiasi di queste fasi, consultare la tabella di ricerca guasti che segue.

ORA E' POSSIBILE INIZIARE AD USARE IL VIC 20

NOTA: E' possibile usare un monitor invece di un televisore – nel qual caso occorre andare direttamente dal VIC al cavo del monitor, senza passare attraverso il modulatore RF.

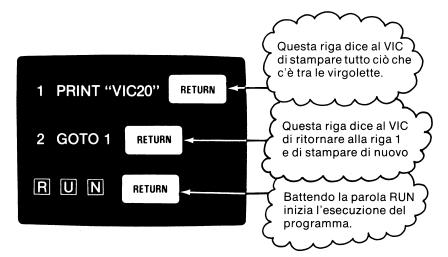
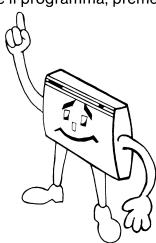
Tabella per la ricerca dei difetti

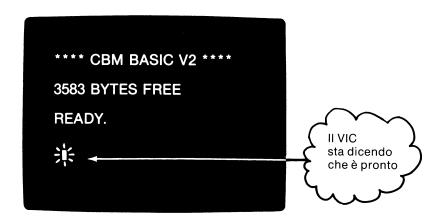
Sintomo	Causa	Rimedio
NESSUNA IMMAGINE (Spia di accensione spenta)	VIC non collegato	Controllare la presa vicino all'interruttore
	Alimentatore non collegato	Controllare il collegamento con la presa
	Fusibile del VIC difettoso	Portare il VIC al centro di Assistenza Autorizzato per la sostituzione del fusibile*
ASSENZA DI IMMAGINE (Spia accesa) (Provare a spegnere il VIC per alcuni secondi quindi riaccenderlo)		
	TV sul canale sbagliato	Controllare il canale per l'immagine
	Collegamento scorretto	Il VIC si collega alla presa UHF sul TV (la presa d'antenna)
	Modulatore non inserito	Controllare il collegamento sul connettore video a 5 piedini
	Cavo video non collegato	Controllare il collegamento sul modulatore
IMMAGINE SENZA COLORE	TV non sintonizzato	Risintonizzare il televisore
IMMAGINE CON COLORI DEBOLI	Regolazione cromatica scorretta sul televisore (vedere «Immagine senza colore»)	Regolare colore/contrastro/luminosità sul televisore
IMMAGINE CON ECCESSIVO RUMORE DI FONDO	Volume televisore troppo alto (vedere «Immagine senza colore»)	Regolare il volume del televisore
IMMAGINE OK MA MANCA IL SUONO	Volume televisore troppo basso	Regolare il volume del televisore

* Il VIC usa un fusibile 3 A a ritardo.

Conoscere il VIC

- Per cominciare
- Il primo programma

Provare a battere questo programma:

Battere questo programma esattamente come indicato e vedere cosa succede!

Per interrompere il programma, premere il tasto .

Per cominciare – Qualche esperimento

E' fatta! Il VIC freme in tutti i suoi colori ed è pronto a dire cosa fare. Il rettangolo lampeggiante blu scuro, detto *cursore,* è il segnale per informare che il VIC è in attesa che si batta qualche cosa.

Suggerimento VIC:

Se si batte involontariamente un carattere indesiderato sullo schermo, premere il tasto

. Questo tasto cancellerà il carattere immediatamente alla sinistra del cursore lampeggiante. Usare questo tasto quanto spesso si desidera per cancellare caratteri indesiderati.

Ora, si parte! Iniziare premendo i seguenti tasti:

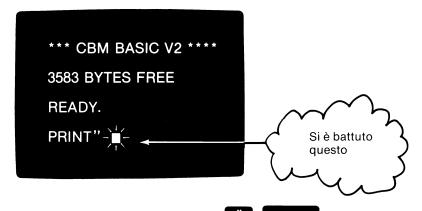
Visto come il *cursore* si muove di una posizione ogni volta che si preme un tasto? Il cursore dice cioè dove apparirà il successivo carattere sullo schermo. OK, ora bisogna trovare il tasto SHIFT, che è simile a quello riprodotto in figura:

SHIFT

Ce ne sono due, il funzionamento è identico.

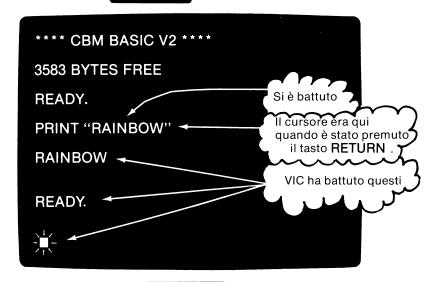
Tenere abbassato il tasto SHIFT e contemporaneamente " 2 :

E' possibile sbloccare il tasto SHIFT dopo aver premuto il tasto " 2 .

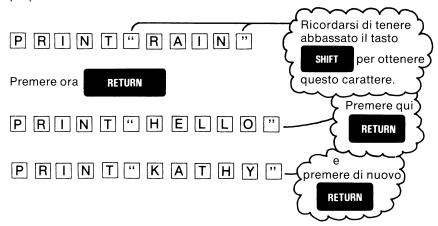
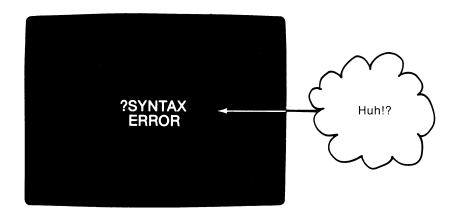
Lo schermo dovrebbe apparire come segue:

La pressione contemporanea dei tasti " 2 e SHIFT ha provocato la comparsa delle virgolette sullo schermo.* Ma continuiamo. Premere ora i seguenti tasti:

R A I N B O W

Infine tenere abbassato il tasto SHIFT e premere di nuovo " 2 .
Lo schermo ora si presenta come segue:

* Nota: Se è comparso il numero 2 sullo schermo invece del segno», significa che non si è tenuto abbassato il tasto SHIFT . Premere una volta INST DEL per cancellare il 2 e provare di nuovo.

La pressione del tasto RETURN dice al VIC che si è terminata la battitura. Il VIC quindi guarda cosa è stato battuto, riconosce che gli è stato chiesto di fare qualche cosa (in effetti stavolta gli è stato detto di *stampare* qualcosa). Il VIC quindi stampa (PRINT) tutto ciò che si trova fra le virgolette (RAINBOW).

Quando il VIC ha finito di stampare la parola RAINBOW, lo fa sapere visualizzando il messaggio READY e facendo lampeggiare il cursore.

Tocca ora all'operatore, che deve immettere qualche altro messaggio PRINT perchè il VIC lo visualizzi. Provare con queste frasi o con altre a propria scelta:

Occorre ora provare con altri caratteri diversi dalle lettere tra i segni di virgolette: il VIC non se ne preoccupa. Nota: Se si compie un errore nel battere la parola PRINT, il VIC ne dà notizia visualizzando questo messaggio sullo schermo:

Non preoccuparsi. *Non c'è assolutamente modo di danneggiare il VIC battendo sulla tastiera* (a meno che naturalmente chi batte non sia un elefante), ma se si compie un errore, il VIC aiuta richiamando l'attenzione sull'errore stesso. Questi messaggi di errore e ciò che essi significano sono spiegati nell'Appendice N. A questo punto non bisogna preoccuparsi del massaggio «SYNTAX ERROR». Basta continuare a fare esperimenti.

In breve lo schermo si riempie con tutto il materiale che si sta battendo. Ma il VIC ha un comodo mezzo per sgombrarlo. Per dire al VIC di «liberare» lo schermo, procedere come segue:

Tenere abbassato il tasto SHIFT e premere il tasto CLR HOME.

Lo schermo si libera istantaneamente; tutto ciò che l'operatore ed il VIC hanno scritto su di esso scompare. Rimane pertanto un'area di visualizzazione bianca pulita ed un cursore blu lampeggiante nell'angolo superiore sinistro.

Ricordarsi cosa si è detto al VIC per fargli liberare lo schermo. La cancellazione dello schermo è uno dei comandi più frequenti che si useranno man mano si imparerà a conoscere il VIC.

Il primo programma

Il VIC si è comportato benissimo nel visualizzare i messaggi per quanto bizzarri potessero essere, quindi il computer è probabilmente pronto a fare qualche altra cosa. Inizieremo «immettendo» il *primo programma di computer.*

FASE 1: Cancellare lo schermo tenendo abbassato il tasto SHIFT e premendo il tasto CLR HOME.

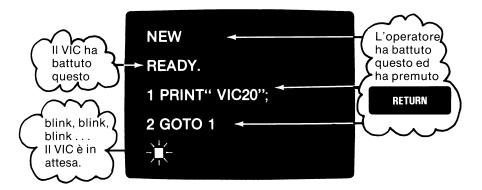
FASE 2: Battere N E W e premere il tasto RETURN.

FASE 3: Battere 1 P R I N T " SPACE V I C 2 0 " ; e premere RETURN

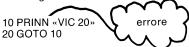
Usare il tasto SHIFT per questi.

FASE 4: Battere: 2 G O T O 1 e premere RETURN

Non battere le lettere di questa parola – basta premere la barra di spazio (il tasto lungo alla base della tastiera del VIC).

Al termine, lo schermo appare come segue:

Suggerimento VIC: Per correggere gli errori in un programma

Se si compie un errore su una riga, ci sono a disposizione queste possibilità di *correzione:*

1. *E' possibile ribattere una riga* in qualsiasi momento. Il VIC automaticamente sostituisce la nuova riga alla vecchia. Per esempio, se il programma appare come segue:

   E' possibile rimediare battendo RETURN alcune volte e quindi battendo:

   10 PRINT «VIC 20» RETURN

   Ora la *nuova riga* ha sostituito la vecchia e il programma «funzionerà». Per assicurarsene, battere [L][I][S][T]. La sostituzione di riga in un programma è anche un rapido e facile modo per *fare esperimenti.*

2. *E' possibile cancellare una riga indesiderata* battendo il *numero* di quella riga e premendo RETURN. L'intera riga verrà cancellata dalla memoria.

3. *E' possibile correggere una riga* usando i tasto che comandano il movimento del cursore per spostare il carattere o i caratteri di una riga del programma che si vuole cambiare, battendo un programma sopra di essi e battendo di nuovo RETURN. Notare che le virgolette talvolta confondono il VIC – se si ottengono caratteri indesiderati dopo le virgolette, tornare all'inizio della riga e ribatterla.

4. Il tasto INST (che si può ottenere battendo SHIFT INST DEL) consente di inserire caratteri aprendo degli spazi in una parola o in una virgola già battute.

5. Il tasto DELETE (basta battere INST DEL) cancella i caratteri posti immediatamente alla sinistra del cursore.

Se tutto sembra corretto, battere la parola seguente e premere RETURN.

Lo schermo dovrebbe riempirsi con tante scritte VIC 20, che sembrano piccole lettere animate che viaggiano sullo schermo.

```
VIC20VIC20VIC20VIC20VIC20VI(
IC20VIC20VIC20VIC20VIC20VIC20'
:20VIC20VIC20VIC20VIC20VIC20V
20VIC20VIC20VIC20VIC20VIC20V
)VIC20VIC20VIC20VIC20VIC20VI
'IC20VIC20VIC20VIC20VIC20VIC20'
C20VIC20VIC20VIC20VIC20VIC20V
20VIC20VIC20VIC20VIC20VIC20V
0VIC20VIC20VIC20VIC20VIC20VI
'IC20VIC20VIC20VIC20VIC20VIC20'
'IC20VIC20VIC20VIC20VIC20VIC20'
'IC20VIC20VIC20VIC20VIC20VIC20'
```

Si vuole rallentare il programma? Basta premere il tasto CTRL posto alla sinistra della tastiera.

Se si tiene abbassato il tasto CTRL il programma *rallenta.*

Interessante! Il VIC è pieno di meravigliose caratteristiche. Qui con una semplice operazione sui tasti si sta dicendo al VIC di ridurre la velocità con cui sta riempiendo lo schermo di caratteri.

Sì, ma come fermare il programma? Buona domanda. Osservare la tastiera finchè non si trova il tasto RUN STOP .

Premere il tasto RUN STOP . Il programma dovrebbe interrompersi e sullo schermo dovrebbe comparire il messaggio:

BREAK IN 1

READY

(non preoccuparsi, non si è rotto il VIC – «break» significa «stop» nel linguaggio che il VIC usa). Inoltre dovrebbe riapparire il cursore. Notato che se n'era andato durante la stampa?

Ora diamo uno sguardo al programma per vedere se c'è ancora. Provare battendo:

Il programma (righe 10 e 20) verrà visualizzato sullo schermo. Battere ora RUN per farlo ripartire.

Sono stati così presentati parecchi aspetti del VIC che si useranno spesso nei capitoli successivi. Finora:

- Si sono stampati (PRINT) messaggi sullo schermo.
- Si è cancellato lo schermo (tasti SHIFT CLR).
- Si è scritto il primo programma (VIC 20) e creato un effetto di animazione.
- Si è rallentato (ConTRoLed) il programma (il tasto CTRL).
- Si è interrotto il programma con il tasto STOP (tasto RUN/STOP).
- Si è listato (LIST) il programma.
- Si è imparato qualche modo facile per correggere ciò che viene battuto erroneamente.

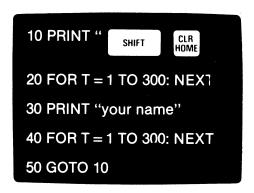
Man mano che si esplorano i capitoli di questa guida si troveranno molti usi di ciò che si è visto finora. Non preoccuparsi se a questo punto ci sono domande che non hanno ancora una risposta. Basta procedere e continuare a *sperimentare:* la maggior parte delle domande ne troveranno automaticamente una.

Questa guide è studiata per consentire di passare direttamente a *qualsiasi capitolo* che sembra interessante. Non occorre cioè leggersi ciascun capitolo nell'ordine per conoscere il VIC. E' però importante partire dall'inizio di ciascun capitolo. Si troverà che questa introduzione graduale a ciascun argomento rende più facile apprendere come creare proprie avventure sullo schermo. E' bene quindi approffitarne!

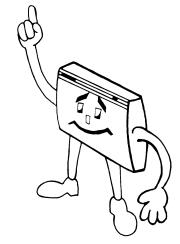
Uso dello schermo e della tastiera

- Il primo carattere grafico
- Un giro della tastiera del VIC 20
- La stampa sullo schermo
- Il calcolatore del VIC 20
- Introduzione al colore

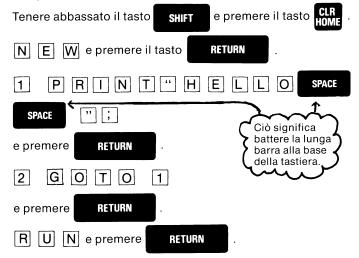
Provare a battere questo programma

Battere questo programma esattamente come indicato e vedere cosa succede!

Battere: N e premere RETURN .

Per interrompere il programma premere il tasto

.

Uso dello schermo e della tastiera

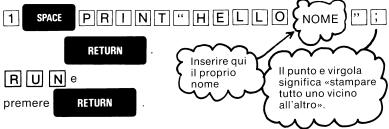
Questo capitolo presume che si sia già letto e compreso il Capitolo 1: in caso contrario occorre tornare indietro a leggere almeno le ultime due sezioni che mostrano come usare la tastiera per controllare ciò che il VIC stampa sullo schermo.

Per iniziare, sedersi davanti alla tastiera del VIC e battere come segue, compreso il numero di riga e di programma ed i segni di punteggiatura:

Quando si batte RUN lo schermo si riempi con la parola:

HELLO

La parola *sembra* muoversi verso l'alto e lateralmente! Premere il tasto CTRL per rallentarne leggermente il movimento. Il VIC sta stampando (PRINT) il messaggio parecchie volte in prossimità del fondo dello schermo. Quando lo schermo si riempie i suoi contenuti vengono spostati *verso l'alto* per far posto ad altri caratteri. Così il movimento verso l'alto sta realmente avvenendo. L'effetto tipico della colonnina girevole del negozio da barbiere è una «illusione» provocata dal numero di caratteri che il VIC sta inserendo su ciascuna riga.

Per interrompere questo programma premere il tasto .

Ora tocca all'operatore. Battere queste due righe:

Vow! Ecco trasformato l'operatore in una stella dello schermo. Cosa fa il nome sullo schermo? L'illusione di movimento dipende da alcuni caratteri presenti nel messaggio.

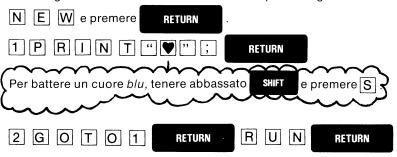
Di nuovo, quando si vuole interrompere l'azione premere il tasto RUN STOP .

Il primo carattere grafico

Cancellare lo schermo (tenere abbassato SHIFT e premere contemporaneamente CLR HOME). Ora battere SHIFT e S. Sullo schermo dovrebbe comparire un *cuore* blu. Provare ancora. Si è così battuto il primo *carattere grafico.*

Cercare ora di battere altri caratteri grafici. Tenere abbassato il tasto C= e battere alcuni caratteri grafici – si tratta di quelli che figurano sul lato sinistro dei tasti. I caratteri grafici di sinistra sono molto utili per disegnare moduli gestionali, tabelle e grafici. Usando contemporaneamente i tasti SHIFT e C= è possibile ottenere lettere minuscole e maiuscole. Per la spiegazione vedere il Capitolo 3.

Dimensione dello schermo

Quanto misura lo schermo del VIC? Vale la pena di provare. Procedere come segue: cancellare lo schermo e battere quanto segue:

Lo schermo si riempirà di cuori blu! Contare il numero di cuori che vengono stampati sullo schermo. Ce ne sono 22 in ciascuna fila. Il VIC ha 22 posizioni di stampa nel senso della larghezza dello schermo, posizioni talvolta dette *colonne.* Il VIC ha cioè 22 colonne.

Quante posizioni ci sono in senso verticale? Premere il tasto CTRL per rallentare la stampa. Abbassando il tasto CTRL si fa in modo che le ultime quattro file lampeggino. Il VIC ha 23 file in senso verticale.

COLONNA

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22

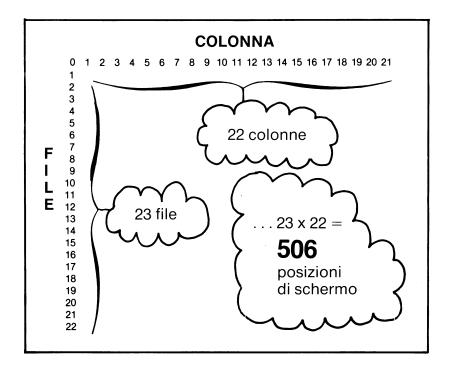
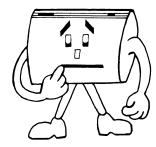
FILE

22 colonne

23 file

. . . 23 x 22 =

506

posizioni
di schermo

Il VIC ha 506 posizioni sullo schermo per i caratteri, le lettere, i simboli e così via. Si potrebbe dire che il VIC può manipolare 506 caratteri alla volta. Interessante!

SUGGERIMENTO VIC

Se il VIC ha 22 colonne, qualsiasi messaggio la cui lunghezza sia un divisore pari di 22 (messaggi di 2, 11 e 22 caratteri), fa sì che il VIC stampi in colonne ordinate. I messaggi di altre lunghezze proseguiranno sulla riga successiva. E' bene verificare questa affermazione.

Si può interrompere la stampa da parte del VIC di ordinate colonne di cuori premendo il tasto RUN STOP. Quindi immettere queste due righe.

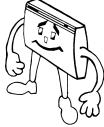
E' possibile cambiare il modo in cui le informazioni sono battute sullo schermo inserendo spazi tra le virgolette. Un altro modo consiste nell'usare i punti invece degli spazi. Provare a battere il proprio nome e 3 punti nel programma all'inizio di questo capitolo.

Un giro della tastiera VIC 20

Finora è stata usata la tastiera per creare e stampare messaggi, inserire caratteri grafici sullo schermo, controllare in flusso di ciò che il VIC sta facendo (CTRL e RUN STOP) ed eventualmente correggere ciò che è stato battuto (INST DEL).

Occorre ora dedicare un po' di tempo per un giro più esteso della tastiera e vedere cosa è in grado di fare. Consultare il diagramma che precede che illustra la potente e versatile serie dei tasti del VIC 20.

Questo è un tasto di «ripristino».

Se si batte il tasto RUN/STOP e contemporaneamente si batte il tasto RESTORE, si ripristina completamente il computer come se lo si fosse appena riacceso . . . con il beneficio che qualsiasi programma presente nella memoria viene conservato e può essere listato o eseguito dall'inizio.

Tasti SHIFT – La tastiera del VIC è esattamente come quella di una macchina da scrivere ed ha due tasti **shift** ed un tasto SHIFT LOCK.
Il tasto SHIFT viene usato con altri tasti per battere caratteri grafici e per eseguire operazioni tipo la cancellazione dello schermo.

Tasto CLR/HOME – La pressione di questo tasto sposta il cursore all'angolo superiore sinistro dello schermo (posizione di partenza).
Se si tiene abbassato il tasto SHIFT e si preme questo CLR/HOME il cursore ritorna in ogni caso alla posizione di partenza ma contemporaneamente cancella lo schermo.

Tasti CRSR – Con il VIC, è possibile spostare facilmente il cursore verso l'alto, verso il basso e lateralmente. I tasti CRSR hanno la capacità di ripetizione automatica, che consente di mantenere il cursore in movimento fino a che non si rilascia il tasto. Ciascun tasto porta una serie di frecce che dicono quali direzioni il tasto controlla – movimento verso l'alto, verso il basso o laterale. Per spostare il cursore verso il basso o verso destra, occorre premere semplicemente l'appropriato tasto. Per spostarlo verso l'alto o verso sinistra occorre tenere abbassato il tasto SHIFT mentre si preme l'appropriato tasto CRSR. E' importante tener presente che è possibile muovere il cursore sopra i caratteri dello schermo *senza influire su tali caratteri.*

Tasto RETURN – Occorre premere RETURN al termine di ciascuna riga di istruzioni. La pressione di questo tasto dice al VIC di immettere la riga o di eseguire l'istruzione o le istruzioni. Talvolta è di aiuto pensare al tasto RETURN come ad un tasto ENTER (di immissione) in quanto questo tasto effettivamente immette le informazioni o le istruzioni nel computer.

Tasto CTRL – Questo tasto viene usato con i tasti COLOR per scegliere i *colori* che si creano sullo schermo del VIC. Il tasto consente inoltre di definire propri comandi *di controllo* che possono essere incorporati in qualsiasi applicazione che è possibile sviluppare per il VIC. Alcune cartucce ad innesto si serviranno del tasto Control per eseguire funzioni speciali.
Il tasto CTRL funziona come il tasto SHIFT. Occorre tenerlo abbassato mentre si preme il tasto del colore.

Tasti dei colori – E' possibile cambiare il colore dei caratteri visualizzati premendo simultaneamente il tasto CTRL ed uno degli 8 tasti colore/ numero sulla riga superiore della tastiera. Sulla superficie anteriore di ciascun tasto c'è un'annotazione stenografica del nome di ciascun colore. I colori sono nero (BLK), bianco (WHT), rosso (RED), blu-verde (CYN), porpora (PUR), verde (GRN), blu (BLU) e giallo (YEL). Con questi tasti è possibile impostare o cambiare i colori delle lettere, dei numeri e grafici visualizzati all'interno o all'esterno del programma di computer. Una volta che si «definisce» il colore, tutto ciò che verrà battuto sarà di quel colore fino a che non lo si cambia di nuovo.

Tasti RVS ON e RVS OFF – E' possibile *invertire* (e cioè portare in negativo) le immagini che il VIC inserisce sullo schermo battendo CTRL e RVS ON. Tutto ciò che viene battuto sarà quindi invertito ossia reso in negativo... ad esempio è possibile far sì che il VIC visualizzi caratteri bianchi su fondo blu (al contrario di ciò che fa normalmente) premendo CTRL e RVS ON. Per ritornare alla presentazione normale, battere CTRL e RVS OFF. Provare!

Tasto RUN-STOP – Premere questo tasto per dire al VIC di *interrompere* l'operazione che sta facendo e riportare il controllo all'operatore. Quando il VIC sta *eseguendo* un programma, è possibile interrompere il programma stesso con questo tasto. Se si tiene abbassato il tasto SHIFT e si preme RUN/STOP, si dice al VIC iniziare a *caricare* le informazioni nella memoria dall'unità a cassette opzionale.

Tasto INST-DEL – Premendo questo tasto è possibile *inserire* e *cancellare* caratteri della riga che si sta battendo. Quando si preme il tasto da solo scompare il carattere che si trova immediatamente alla sinistra del cursore. Se si è nel mezzo di una riga, il carattere a sinistra viene cancellato ed il carattere alla destra automaticamente si sposta verso sinistra per occuparne lo spazio. Se si tiene abbassato SHIFT e si preme questo tasto, si *crea* uno spazio nella riga in modo da potervi *inserire* un nuovo carattere. Ciò è molto utile per correggere ed eliminare errori!

Grafici e tasto del COMMODORE – Quando si accende il VIC, si è automaticamente nel modo «grafici» il che significa che è possibile battere LETTERE MAIUSCOLE e gli oltre 60 segni grafici che si vedono sui tasti. Su ciascun tasto ci sono due segni grafici. Per ottenere quello sul lato di destra, basta tenere abbassato il tasto SHIFT e battere il tasto con il segno desiderato.
Per ottenere il segno grafico sul lato sinistro, tenere abbassato il tasto «COMMODORE» (la piccola bandierina). In questo modo è possibile battere contemporaneamente LETTERE MAIUSCOLE e l'intera serie di grafici! E' possibile creare immagini, tabelle e disegni inserendo i caratteri fianco a fianco o uno sopra l'altro (come mattoni).

Tasti MAIUSCOLO/MINUSCOLO e tasti per GRAFICI – Se si premono contemporaneamente i tasti SHIFT e COMMODORE si predispone il video nel modo *testo*. E' possibile quindi usare il VIC come una normale macchina da scrivere, con lettere maiuscole e minuscole più tutti i segni grafici posti sul *lato sinistro* dei tasti. I segni grafici sul lato sinistro sono l'ideale per creare tabelle, grafici e moduli gestionali. Per ritornare al modo «maiuscole/grafici» premere insieme i tasti SHIFT e COMMODORE.

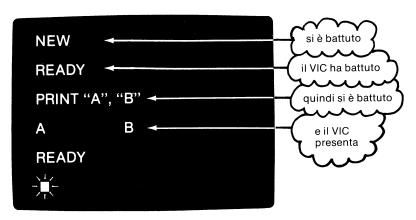
Tasti di funzione programmabili – I quattro tasti marrone sul lato sinistro della console non sono definiti quando si accende il VIC. Ad essi possono essere assegnate *funzioni* dall'interno delle applicazioni che l'operatore stesso crea. Usando questi tasti con e senza SHIFT, è possibile ottenere un totale di otto tasti di funzione definibili dall'utente. I tasti di funzione saranno prevalentemente usati con le cartucce ad innesto contenenti programmi speciali ma possono anche servire egregiamente ai programmatori.

Tasti speciali – La tasteria del VIC contiene anche simboli speciali che non si trovano su molte macchine da scrivere o addirittura sulla maggior parte dei computer. Ad esempio figurano il simbolo della sterlina (£), il pi greco (π), la freccia verso sinistra (←) la freccia verso l'alto (↑), il segno di maggiore/minore (><) e le parentesi quadre ([]).

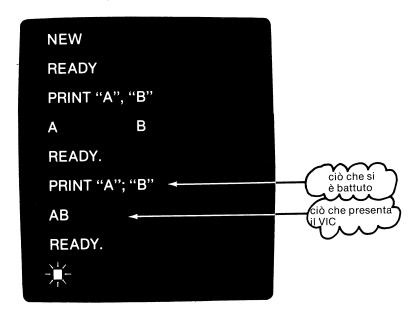
E con ciò si conclude il giro della tastiera del VIC. Usando le sole parole è difficile spiegare quanto flessibile e potente sia questa tastiera. Il modo migliore per scoprirlo è di iniziare un giro da soli. Fare esperimenti con la tastiera. Provare le varie funzioni maiuscolo/minuscolo sopra indicate. Vedere ciò che è possibile creare con la ricca serie di caretteri grafici del VIC. La tastiera è il collegamento diretto con il VIC. Conoscere la tastiera significa automaticamente conoscere il VIC 20.

Cancellare lo schermo ed immettere le righe seguenti:

NEW RETURN PRINT «A», «B» RETURN

Sullo schermo compare:

Ora immettere questa riga e premere RETURN :

PRINT «A»; «B»

Lo schermo ora presenta:

Quando si è usata la virgola nella prima istruzione PRINT, il VIC ha inserito le lettere sullo schermo separandole con parecchi spazi. Quando è stato usato il punto e virgola, il VIC ha visualizzato le due lettere l'una vicina all'altra.

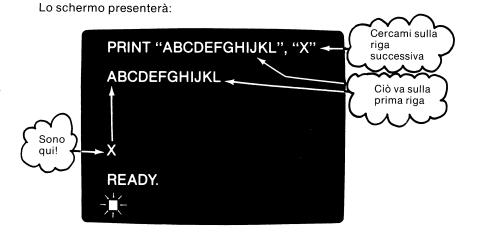
Nel primo caso le lettere sono distanziata esattamente di 11 spazi. Questo fatto dà un'idea di ciò che sta succedendo. Il VIC divide l'area dello schermo in due parti uguali.

Quando il VIC sta stampando (PRINT) due messaggi o numeri separati da una *virgola,* inserisce il primo sul lato sinistro dello schermo e il seconda su quello di destra . . .

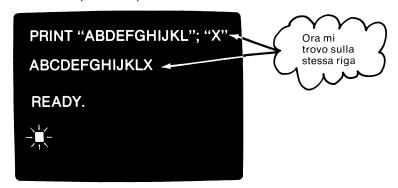
Il *primo* non sia lungo più di 11 caratteri.

Se il primo è minore di (o uguale a) 11 caratteri, il VIC lo stampa e quindi si sposta al centro dello schermo per visualizzare il secondo. Se il primoè più lungo di 11 caratteri, il secondo appare sulla riga seguente. Cancellare lo schermo e provare con questo esempio:

PRINT «ABCDEFGHIJKL», «X»

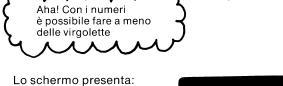
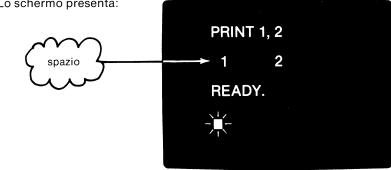
Lo schermo presenterà:

La prima parte del messaggio è lunga 12 caratteri cosicchè la «X» termina alla riga successiva. Ripetere questo esempio con un punto e virgola (;) tra i due elementi.

PRINT «ABCDEFGHIJKL»; «X»

Lo schermo presenta questo risultato?

Capito il concetto? Il VIC agisce come una macchina da scrivere con una *tabulazione* automatica in prossimità del centro dello schermo. Quando vede la virgola, si sposta al centro dello schermo o all'inizio della riga successiva, quello dei due che è più vicino.

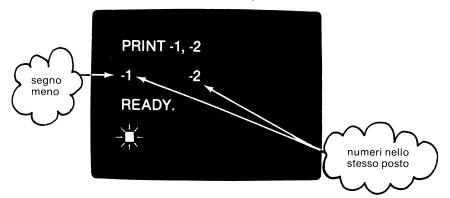
Cancellare lo schermo e battere la seguente riga sul VIC:

PRINT 1, 2

Lo schermo presenta:

Visto lo spazio davanti al primo numero? Quando il VIC visualizza i numeri, lascia uno spazio all'inizio per il *segno* del numero. Se il numero è positivo se vede uno spazio vuoto. Se il numero è negativo sullo schermo dovrebbe comparire un *segno meno* (–).

Provare per credere. Immettere questa riga sul VIC:

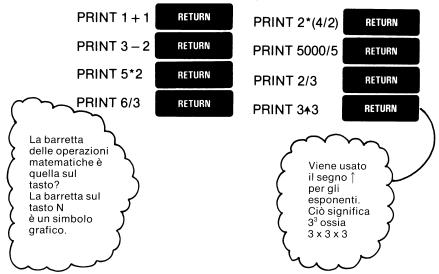
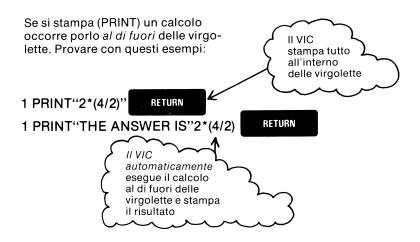
PRINT –1, –2

Osservare lo schermo e vedere cosa compare.

I numeri appaiono allo stessò posto dell'esempio precedente; ora sono però preceduti dal segno meno (–).

Questi pochi esempi danno qualche idea del modo in cui il VIC può aiutare ad inserire messaggi ed informazioni sullo schermo. Il VIC ha molti altri modi per facilitare in questa funzione e sarà facile impararli continuando ad usarlo.

Il calcolatore del VIC

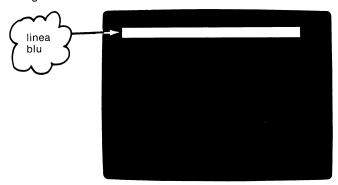
Il VIC può anche essere usato come calcolatore a nove cifre. I segni più e meno sono usati esattamente come nelle operazioni aritmetiche.
Il segno di moltiplicazione del VIC è l'asterisco (*) e il segno di divisione è la barretta (/). Battere questi calcoli e controllare i risultati. Vedere le Appendici C e K per ulteriori informazioni.

Se si stampa (PRINT) un calcolo occorre porlo *al di fuori* delle virgolette. Provare con questi esempi:

Introduzione al colore

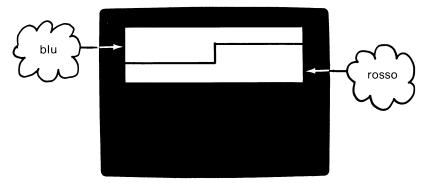
Il VIC può battere lettere, numeri e simboli grafici in otto colori diversi. Esso può inoltre stampare caratteri in *negativo.*

Con lo schermo vuoto, tenere abbassato il tasto CTRL e premere questo tasto:

RVS ON

Ora lasciare lo spazio CTRL e premere la barra SPACE posta alla base della tastiera. Tenere abbassata la barra SPACE. Cosa succede? Viene disegnata una linea blu attraverso lo schermo?

Tenere abbassata la barra di spazio finchè lo si desidera. Quando il cursore scompare sul lato destro dello schermo, ricompare su quello sinistro e la linea blu inizia a formare una barra colorata blu più grande.

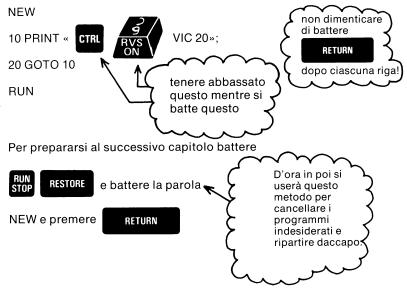
Rilasciare la barra SPACE e tenere: abbassato il tasto CTRL premendo contemporaneamente il tasto RED.

Il cursore dovrebbe ora essere *rosso.* Premere e tenere abbassato la barra SPACE ancora una volta. Inizia a formarsi una nuova barra di color rosso? Sì! Bene, continuare a dipingere!

E' possibile passare ad altri colori a piacere. E' possibile inoltre creare barre colorate spesse o sottili. E' interessante godersi questa abilità di recente scoperta del VIC che dà un po' di colore alla vita.

Battere ora CTRL RVS OFF e premere la barra di spazio.

Non succede nulla tranne che si creano spazi vuoti.

Battere CTRL RVS ON.

e ricompare la barra colorata. Cercare di battere qualche lettere in negativo. Le lettere in negativo servono bene per battere titoli e sono spesso usate per evidenziare parole o numeri speciali. E' possibile inoltre usare caratteri negativi all'interno di un programma. Per esempio, provare con questo:

Colore e grafici

- Programmazione a colori
- I tasti colore del VIC
- Come cambiare i colori dei margini e dello schermo
- Combinazioni di colori dei margini e dello schermo
- Colorazione dello schermo
- Posizioni dello schermo
- Colori casuali
- Combinazioni di suono e di colore
- I grafici da tastiera
- I grafici nei titoli

Provare a battere questo programma:

Battere questo programma esattamente come indicato e vedere cosa succede!

Per interrompere il programma premere il tasto RUN STOP .

Colore e grafici

Il VIC unitamente ad un televisore a colori può consentire di inserire colori ovunque sullo schermo. Quando si accende il VIC, i margini, il cursore e qualsiasi carattere sullo schermo saranno già *a colori.*
Ma questo è solo l'inizio. Il VIC può visualizzare otto colori del cursore, otto colori dei margini e 16 colori dello schermo!

Usare i tasti sulla tastiera del VIC per far apparire il colori sullo schermo. Battere qualsiasi lettera. La lettera dovrebbe comparire in blu scuro su fondo bianco. Provare ora la fila superiore di tasti (tasti numerati da 1 a 8). Guardando la parte inferiore si possono notare i nomi (abbreviati – in inglese) dei vari colori. Trovare il tasto contrassegnato CTRL sul lato sinistro della tastiera.

Tenere abbassato il tasto CTRL e battere il tasto contrassegnato YEL.

Rilasciare il tasto CTRL ed il tasto YEL e battere qualsiasi lettera sulla tastiera. Questa lettera dovrebbe apparire in giallo. Ora tenere abbassato il tasto CTRL e battere un altro colore, quindi battere alcune lettere. Visto come è facile cambiare il colore delle lettere sullo schermo?

Provare ora il tasto contrassegnato RVS ON. Tenere abbassato il tasto CTRL e battere il tasto RVS ON. Provare a battere qualche altra lettera. Tutte le lettere battute (fino a che non si preme il tasto RETURN) comparianno in *negativo* sullo schermo, come in un negativo fotografico. Se si tiene abbassato il tasto CTRL e si batte RVS OFF, le lettere verranno visualizzate normalmente.

Programmazione a colori

Ora combineremo il controllo del colore con un semplice comando di programma. Notare che quando si batte CTRL ed un tasto di colore all'interno delle virgolette, compare un simbolo grafico in negativo. Ciò è del tutto regolare. Procedere come segue:

Tenere abbassato il tasto SHIFT e premere il tasto CLR HOME.

Battere le lettere:

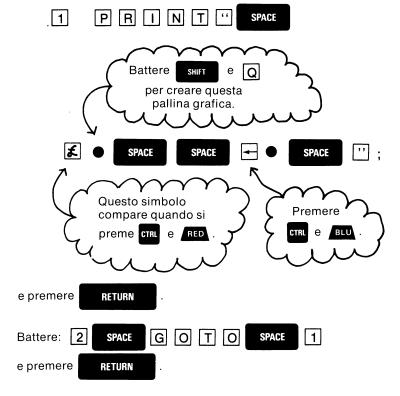
N E W e premere il tasto RETURN

Battere quindi:

1 P R I N T " SPACE

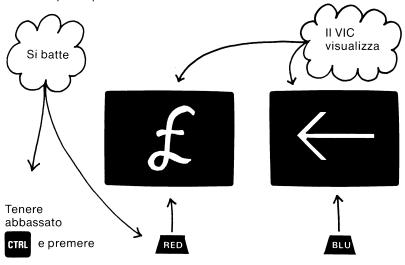
e premere RETURN.

Battere: 2 SPACE G O T O SPACE 1

e premere RETURN.

Se si hanno dei problemi nel battere questo esempio, ritornare alla parte intitolata «Per cominciare» nel Capitolo 1. Ricordarsi comunque che non è possibile danneggiare il VIC in alcun modo, qualsiasi cosa si batta ma è possibile confondersi con talune combinazioni di tasti. Se accidentalmente si batte il tasto SHIFT LOCK, per esempio, lo schermo risultante è difficile da decifrare. Se si fa un errore, battere alcune volte RETURN e ribattere l'intera riga. La nuova riga sostituirà automaticamente la vecchia. Quando si hanno le due righe sopra indicate sullo schermo, battere:

R U N e premere RETURN .

Non appena si preme il RETURN finale, si dovrebbero vedere centinaia di palline rosse e blu galleggiare sullo schermo. Come? E' facile con il VIC. Occorre però tornare all'esempio precedente. Visti i due strani caratteri nella riga che inizia con «1»? Essi sono stati creati quando si è abbassato il tasto CTRL e si sono premuti i tasti RED e BLU.

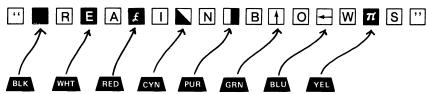
I simboli che compaiono sono la stenografica che il VIC usa per dire di creare la prima pallina rossa e la seconda blu.

Quando si è stanchi di vedere palline rosse e blu, premere STOP. Chi apprezza l'esplorazione in forma libera può riprovare a ribattere la riga 1.

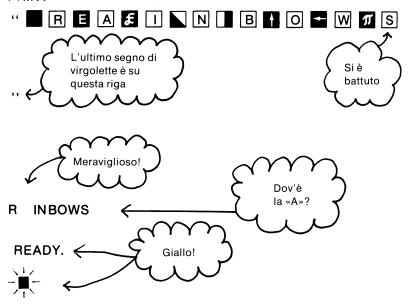
Inserire alcuni CTRL ed alcuni tasti di colore unitamente con caratteri grafici, lettere o numeri. Il VIC può elaborarli tutti e fornire una rappresentazione a colori interessante.

I tasti colore del VIC

Nell'ultimo esempio si è scoperto che è possibile inserire comandi dei colori in un messaggio PRINT usando il tasto CTRL e quelli le cui superfici anteriori sono contrassegnate come segue:

Questi tasti sono contrassegnati con i numeri da 1 a 8. Quando si preme uno di questi tasti nel mezzo di un'istruzione PRINT, compare una serie di caratteri «strani». Per vedere di che cosa si tratta, premere RUN STOP (se il programma è ancora in esecuzione), cancellare lo schermo ed immettere:

PRINT

Non dimenticare di inserire le virgolette o di usare il tasto CTRL con i tasti dei colori. Lo schermo dovrebbe presentare quanto segue:

PRINT

Dov'è la lettera «A»? Oh! Il secondo tasto dei colori è WHT e il fondo è . . . provare ad indovinare . . . *bianco.*

La stampa (PRINT) di una lettera bianca «A» su un fondo bianco crea uno spazio nei RAINBOWS (arcobaleni).

Le altre sette lettere nella parola RAINBOWS rappresentano ciascuna un colore diverso. L'ultima lettera, S, è gialla. Notare che anche il messaggio READY (pronto) è in giallo unitamente al cursore. *Quando i comandi dei colori sono inseriti nel messaggio PRINT, il VIC ricorda l'ultimo colore usato e continua ad utilizzarlo.*

Per riportare il cursore al normale colore blu, tenere abbassato CTRL e premere BLU . Se lo si desidera, provare a stampare (PRINT) alcuni messaggi colorati a scelta. Si avrà così la possibilità di vedere altri usi dei tasti dei colori in brevissimo tempo. Ma ora c'è un annuncio importante . . .

EXTRA!! EXTRA!!

Il Color Show VIC è in arrivo!!

Come cambiare i colori dei margini e dello schermo

Ora che si sa come cambiare il colore delle lettere e dei segni grafici, si vedrà come cambiare i colori dello schermo e dei margini. E' possibile visualizzare 255 diverse combinazioni di colori per il margine e per lo schermo. Per assicurarsi che il VIC è pronto per la prossima operazione, premere RUN STOP e quindi SHIFT , CLR HOME per cancellare lo schermo. Successivamente battere queste righe:

N E W e premere RETURN .

(Premere RETURN al termine di ciascuna delle righe seguenti).

Il carattere più a sinistra su ciascuna riga

```
1FOR X = 1 TO 255
2POKE 36879, X
3PRINT " ♥ POKE 36879," X
4FOR T = 1 TO 1000: NEXT T
5NEXT X
```

Inserire cosa?!

Per questo tenere abbassato SHIFT e premere CLR HOME

Aggiungere uno a «X» – in altre parole cambiare «X» nel successivo numero e ripetere di nuovo

Questo è un «ritardo» che il VIC fa durare da 1 a 1000 (whew!) tra le modifiche

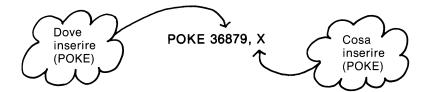
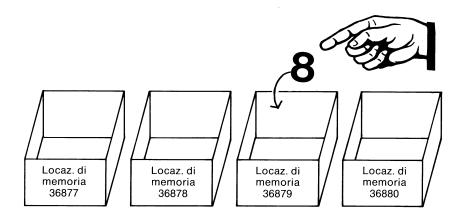
Osservare queste righe una volta battute e vedere se corrispondono a ciò che è indicato su questa pagina. Se ce ne sono alcune che non corrispondono, ribatterle dall'inizio. Usare il tasto CRSR per ritornare su qualsiasi riga che si vuole ribattere.

Una volta soddisfatti in quanto tutte le righe sono corrette, prendere un pezzo di carta ed una penna o matita e sistemarseli accanto. Quindi battere:

Lo schermo dovrebbe incomiciare ad arrossire e a lampeggiare. Il margine cambia colore. Il fondo cambia colore. Anche il piccolo messaggio nella parte superiore dello schermo sta cambiando i suoi colori. Il VIC sta cioè visualizzando 255 diverse combinazioni di colori. (C'è un errore di sintassi in una delle righe? Ribattere quella riga e quindi battere:

Mentre il VIC sta lavorando, se si nota una particolare serie di colori che sembra interessante, annotare il numero di quella combinazione. Cambia soltanto il numero al termine del messaggio stampato. La scritta «POKE 36879» rimane inalterata. (Di tanto in tanto non sarà possibile leggere il messaggio in quanto è nello stesso colore del fondo. Può succedere! Le lettere ed i numeri appariranno dopo alcuni lampeggi ulteriori). Il VIC dispone di otto colori per il margine, di 16 colori per il fondo e di otto colori per i caratteri. E' possibile inserire i caratteri di tutti gli otto colori su qualsiasi fondo. Ciò consente di esplorare numerosissime combinazioni.

Se si vuole riportare lo schermo si colori originali, basta tenere abbassato il tasto RUN STOP e battere RESTORE.

La riga due nell'esempio suddetto è responsabile del cambiamento dei colori del VIC. La riga contiene un comando POKE. Ogni comando POKE ha due valori numerici che il VIC usa:

Il primo numero (in questo caso 36879) è la locazione di memoria (pensare ad una piccola casella contrassegnata 36879) nella quale si va ad inserire (POKE) il secondo numero, X. La locazione di memoria 36879 è il punto in cui il VIC memorizza la sua informazione relativa a quali devono essere i colori del margine e del fondo. Ciascun valore di «X» corrisponde ad una diversa combinazione di colori che il VIC può visualizzare. Per questo esempio, «X» inizia con il valore 1, quindi passa a 2, 3, e così via fino ad un valore di 255.

UN ESEMPIO DI POKE 36879, 8

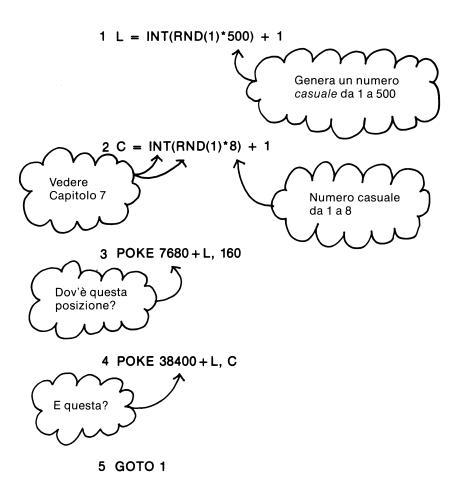
Per facilitare in questa ricerca delle combinazioni di colori *perfette,* ecco la tabella dei valori POKE (valori «X») e dei colori del margine e del fondo corrispondenti. La tabella fornisce i valori POKE che producono tutte le combinazioni di questi colori. I valori POKE sono in sequenza con incrementi di otto. I numeri mancanti sono i valori POKE che provocano *l'inversione* e cioè la presentazione in negativo dei caratteri visualizzati.

Combinazione dei colori del (Margine e dello schermo)

	Margine							
Schermo	BLK (Nero)	WHT (Bianco)	RED (Rosso)	CYAN (Blu-Verde)	PUR (Porp.)	GRN (Verde)	BLU (Blu)	YEL (Giallo)
NERO	8	9	10	11	12	13	14	15
BIANCO	24	25	26	27	28	29	30	31
ROSSO	40	41	42	43	44	45	46	47
BLU-VERDE	56	57	58	59	60	61	62	63
PORPORA	72	73	74	75	76	77	78	79
VERDE	88	89	90	91	92	93	94	95
BLU	104	105	106	107	108	109	110	111
GIALLO	120	121	122	123	124	125	126	127
ARANCIO	136	137	138	139	140	141	142	143
ARAN. CHIARO	152	153	154	155	156	157	158	159
ROSA	168	169	170	171	172	173	174	175
AZZURRO CHI.	184	185	186	187	188	189	190	191
PORPORA CHI.	200	201	202	203	204	205	206	207
VERDE CHI.	216	217	218	219	220	221	222	223
AZZURRO	232	233	234	235	236	237	238	239
GIALLO CHI.	248	249	250	251	252	253	254	255

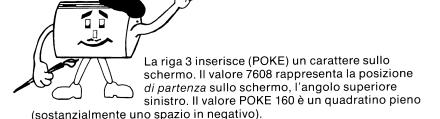
Colorazione dello schermo

A questo punto è bene prendere tempo per dedicarsi a combinare i colori, i caratteri ed i grafici del VIC in una gradevole visualizzazione cromatica. Basta soltanto immettere alcune righe nel VIC per vedere comparire eventi spettacolari.

Se alcuni esempi stanno ancora rimbalzando o agitandosi sullo schermo, occorre interromperli battendo RUN STOP. Cancellare lo schermo. Quindi battere queste righe misteriose.

Controllare l'esempio per accertarsi che tutte le righe siano battute come indicato. Se sono tutte corrette, cancellare lo schermo e battere R U N. Lo schermo dovrebbe ora esplodere in una miriade di colori.

Il VIC ha colpito ancora! Con solo alcune operazioni sui tasti sta ora spostando colori dappertutto. Cosa è successo?

In termini semplici, la riga 1 e 2 scelgono casualmente dove finirà il colore sullo schermo (L sta probabilmente per Location = posizione) e di quale colore si tratterà (C per *colore*). La riga 1 genera da 1 a 500 numeri. La riga 2 genera un numero da 1 a 8.

La riga 3 inserisce (POKE) un carattere sullo schermo. Il valore 7608 rappresenta la posizione *di partenza* sullo schermo, l'angolo superiore sinistro. Il valore POKE 160 è un quadratino pieno (sostanzialmente uno spazio in negativo).

La riga 4 inserisce (POKE) il colore sul carattere. Il valore 38400 rappresenta la locazione di memoria del componente cromatico del carattere in posizione *di partenza.*

Posizioni dello schermo

SUGGERIMENTO VIC:

Per inserire (POKE) caratteri sullo schermo, occorre inserire (POKE) la posizione dello schermo ed il colore di quella posizione per ciascun carattere. Le posizioni dello schermo iniziano a 7680. Le posizioni dei colori iniziano a 38400. Vedere l'Appendice per la mappa di memoria dello schermo.

Occorre a questo punto cercare di inserire (POKE) altri valori sullo schermo.
Cambiare il 160 nella riga 3 in qualsiasi numero compreso fra 0 e 255.

PROVARE – ed ammirare la personalità piena di colori del VIC.

Ci sono 506 possibili posizione sullo schermo (tabella). E' possibile scrivere sullo schermo qualsiasi parola, lettera, frase, segno grafico o qualsiasi cosa, ogniqualvolta lo si desidera. Basti pensare che lo schermo si compone di 506 caselle come questa – ciascuna delle quali ha un proprio numero.

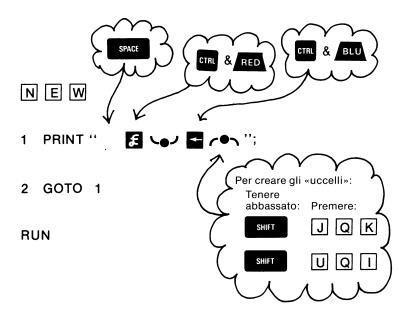
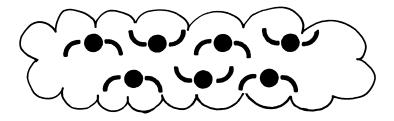
Sotto con gli esperimenti! Ma è ora il momento di ritornare ai tasti di controllo del colore.

SUGGERIMENTO VIC:

Per riportare il VIC ai normali colori di margine e dello schermo, battere questa istruzioni POKE:

POKE 36879, 27 RETURN

Colori casuali

Si cercherà ora di capire come si può fare in modo che il VIC scelga i colori da inserire sullo schermo. Cancellare innanzitutto lo schermo e battere queste righe (premere RETURN dopo ciascuna riga):

Lo schermo dovrebbe riempirsi di uccelli blu e rossi. Indovinate adesso dove stanno volando? Perchè si muovono effettivamente!

Ma lasciamoli volare per un po'. Per interromperne il volo, premere .

Ora che si sa come creare uccelli, cancellare lo schermo e battere queste righe: (Ricordarsi di battere RETURN al termine di ciascuna riga).

N E W

1 A$ = " ■ E £ ◣ ▐ ↑ ← π "

Per battere questa riga, tenere abbassato CTRL e premere ciascuno dei tasti dei colori (BLK WHT RED CYN PUR GRN BLU YEL).

2 N = INT(RND(1)*8) + 1

Questa riga fornisce numeri interi *casuali* da 1 a 8

3 B$ = MID$(A$,N,1)

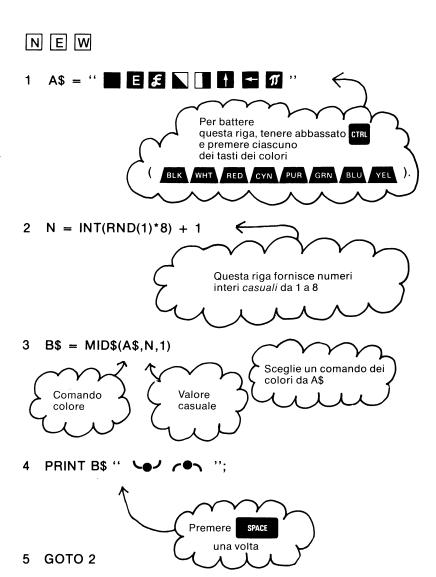
Comando colore

Valore casuale

Sceglie un comando dei colori da A$

4 PRINT B$ " ╰●╯ ╭●╮ ";

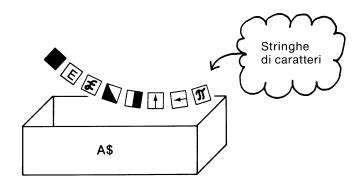
Premere SPACE una volta

5 GOTO 2

Osservare le righe non appena vengono immesse. Accertarsi che corrispondano a quanto qui indicato. Una volta pronti, cancellare lo schermo e battere [R] [U] [N]. Se ci sono errori, ribattere l'intera riga e battere di nuovo [R] [U] [N]. Fallo ancora SAM (scusa, VIC!!). Uccelli! Uccelli! Ovunque!! In ogni colore dell'arcobaleno! Lo schermo dovrebbe riempirsi di stormi di uccelli «arcobaleno». Notare che ci sono degli spazi vuoti nello stormo. E questo perchè in quegli spazi il VIC sta visualizzando uccelli bianchi. Se si vuole che questi uccelli invisibili compaiano, occorre cambiare il colore del fondo. (Come? Occorre inserire – POKE – un numero nella locazione 36879).

Naturalmente se si cambia lo sfondo in un altro colore, ad esempio blu, compariranno gli uccelli blu. Provare per credere. Premere [RUN STOP] quando si vuole far smettere agli uccelli di volare.

L'ultimo esempio ha usato qualche carattere interessante del VIC per far comparire gli uccelli sullo schermo e per cambiarne il colore. Vediamo l'esempio riga per riga. Sulla prima riga:

1 A$ = " [BLK] [E] [£] [◣] [◨] [↑] [←] [π] "

I comandi dei colori formano una lunga *stringa* di caratteri e sono inseriti nella memoria del VIC in una posizione denominata A$. Si può pensare a A$ come ad una scatola in cui possono essere inserite stringhe di caratteri. La prima posizione della scatola A$ contiene il carattere di comando [BLK]. L'ultima posizione contiene il carattere di comando [YEL].

La riga successiva è:

2 N = INT(RND(1)*8) + 1

Questa riga genera un numero intero positivo *casuale* da 1 a 8 ed inserisce quel numero in «N». «N» è il posto nella memoria del VIC che può essere usato per memorizzare i numeri. Il VIC sa che «N» è un posto in cui memorizzare i numeri dato che al termine del nome non c'è il segno del dollaro ($).

Nella riga uno, la locazione usata per memorizzare i comandi dei colori era denominata A$ e presentava il segno del dollaro al termine. Il VIC sa che le locazioni i cui nomi terminano con il segno sel dollaro devono essere usate per memorizzare messaggi o stringhe di caratteri.

Per ulteriori informazioni sul modo in cui

[I] [N] [T] e [R] [N] [D] funzionano insieme per generare numeri casuali, vedere il Capitolo 7 di questo manuale. Per ora basta sapere che questa riga sta producendo numeri da 1 a 8. Visto il numero 8 sul lato destro del segno di uguale? Quel numero stabilisce quanti numeri casuali vengono generati. Se si dovesse cambiare il numero da otto a sei, verrebbe prodotto un qualsiasi numero da uno a sei.

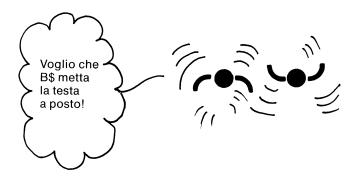
Hmmm . . . ed ora procediamo ulteriormente nell'analisi:

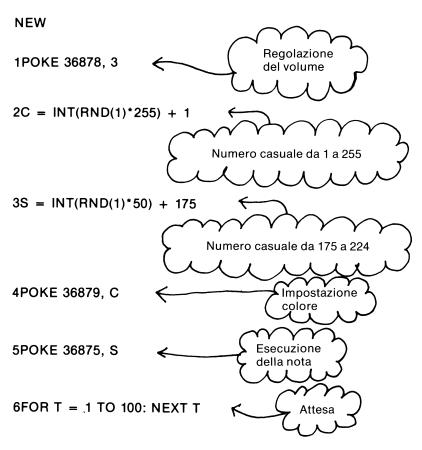
3 B$ = MID$(A$,N,1)

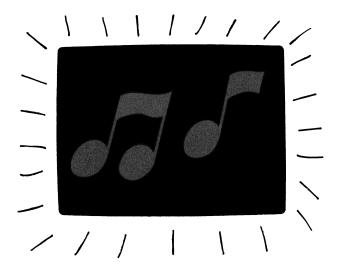
Questa riga crea una nuova «casella» denominata B$ ed inserisce un carattere di controllo. A$ contiene gli otto caratteri di controllo del colore. «N» è un numero casuale da uno a otto. Cosa fa MID$? Non fa altro che scegliere un carattere di controllo colore da A$ nella posizione «N-esima» nella stringa. Questo carattere viene inserito in B$. Sì, ora ci sono tre piccole caselle all'interno del VIC, una denominata A$, una denominata «N» ed una denominata B$. Ogniqualvolta che il VIC arriva alla riga due, viene attribuito un nuovo valore a «N». Ciò determina quale colore MID$ preleva ed inserisce in B$.

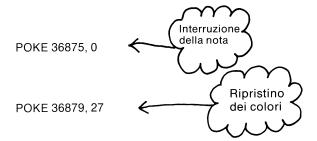
La quarta riga:

4 PRINT B$ " [graphic characters] ";

Questa riga dice al VIC di usare il colore che trova nella casella B$ e disegnare con esso le creature simili ad uccelli. Il comando PRINT B$ crea lo stesso effetto che si avrebbe battendo un carattere di controllo colore specifico. Usando B$, comunque, è possibile fare in maniera che il VIC cambi il carattere del colore automaticamente. Quando «N» (riga 2) cambia, cambia anche B$ (riga 3) e cambiano pure i colori degli uccelli.

La capacità del VIC di generare numeri casuali può essere combinata con le sue caratteristiche grafiche, sonore e cromatiche in molti modi interessanti. Per esempio, possiamo rivisitare il Color Show del VIC ma casualmente e produrre contemporaneamente qualche rumore.

In questo ultimo esempio colori e suoni del VIC sono prodotti entrambi casualmente. Immettere le righe seguenti nel computer:

```
NEW

1POKE 36878, 3

2C = INT(RND(1)*255) + 1

3S = INT(RND(1)*50) + 175

4POKE 36879, C

5POKE 36875, S

6FOR T = 1 TO 100: NEXT T

7GOTO 2
```

Le didascalie nei fumetti spiegano a sufficienza questa esempio. Il Capitolo 5 si addentra più in profondità a proposito della musica. Se si è pronti ad eseguire l'esempio, cancellare lo schermo e battere [R] [U] [N] quindi premere RETURN .

Che meravigliosa combinazione di suoni e di colori! Naturalmente non è possibile aspettarsi che la musica casuale sia adatta ai gusti di tutti.

Quando si vuole fermare il funzionamento di questa macchina dei colori e dei suoni, premere RUN STOP. Occorre quindi battere queste due righe per interrompere la nota che sta ancora suonando e ripristinare lo schermo del VIC ai suoi normali colori.

Ora è opportuno dedicare un poco di tempo a fare esperimenti. Provare i vari colori del VIC. Provare ad ottenere colori e musica.

I grafici da tastiera

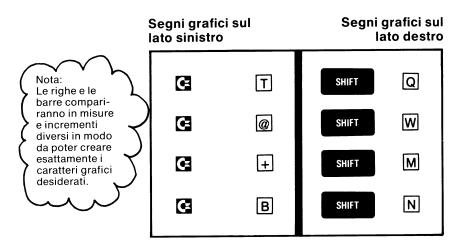
Una delle caratteristiche speciali del VIC 20 è la serie di tasti con i segni grafici. La maggior parte dei tasti rappresentano due caratteri grafici nella parte anteriore. E' possibile visualizzare questi segni grafici sullo schermo, oppure se si dispone di una stampante ad aghi Commodore, stamparli su carta unitamente a tutti gli altri simboli della tastiera.

Per visualizzare il segno grafico sul lato sinistro del tasto, tenere abbassato il tasto Commodore mentre si preme il tasto contenente il segno grafico. Per visualizzare il segno grafico alla destra del tasto, tenere abbassato il tasto SHIFT e premere il tasto desiderato.

SUGGERIMENTO VIC:

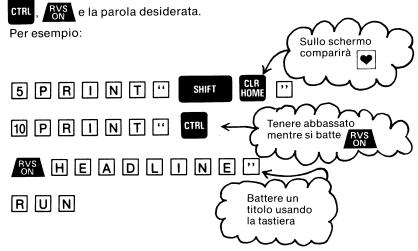
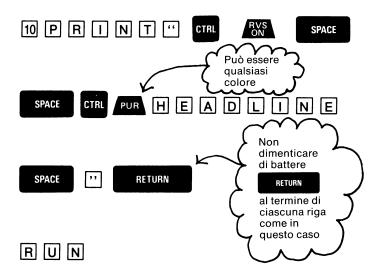
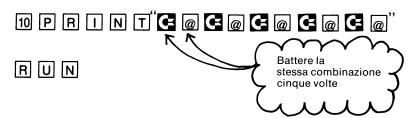
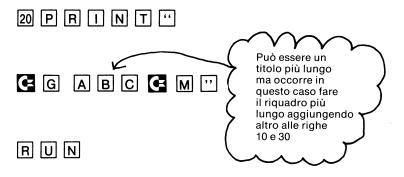
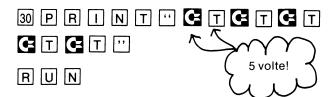
Attenzione . . . se si premono contemporaneamente i tasti SHIFT e Commodore si passa immediatamente al modo maiuscole/minuscole in cui sono disponibili soltanto i segni grafici del lato sinistro. Per ritornare al modo per grafici precedente, premere di nuovo SHIFT e C= contemporaneamente.

Il modo più facile per usare i segni grafici del VIC consiste nel battere il tasto COMMODORE C= oppure SHIFT unitamente al simbolo grafico che si vuole visualizzare dalla tastiera. Per esempio, battere quanto segue:

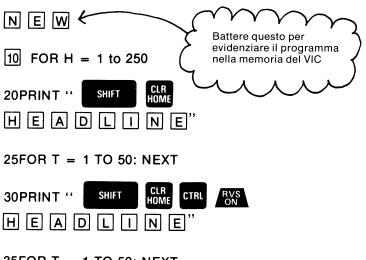
Si dovrebbe vedere comparire sullo schermo un *cuore blu*. Provare a battere qualche altro tasto con segni grafici. Eccone alcuni che è possibile provare:

Notare che i segni grafici sul lato destro sono utili per creare tabelle, diagrammi e moduli gestionali. I segni grafici sul lato destro sono utili per creare illustrazioni, effetti di animazione . . . addirittura carte da gioco!

I grafici nei titoli

I segni grafici non sono limitati ai disegni animati ed ai giochi.
E' possibile usare numerosi effetti speciali per mettere in evidenza titoli, tabella o diagrammi o evidenziare parole speciali nei programmi che contengono molto testo. Il modo più facile per evidenziare una parola o una frase consiste nel batterla in NEGATIVO. Battere semplicemente CTRL, RVS ON e la parola desiderata.

Per esempio:

Provare ora con alcuni spazi per creare una barra di titolo . . .

Un altro modo per evidenziare una parola consiste nel disegnarle intorno un riquadro. La tecnica è come quella del disegno. Assicurarsi di battere esattamente come indicato.

Lo schermo presenta una linea retta. La riga successiva comprende il titolo ed una riga verticale a ciascuna estremità per completare i lati del «riquadro».

Visto come si fa a costruire il riquadro del titolo? Ora per finire . . .

SUGGERIMENTO VIC:

Per correggere il programma battere LIST e premere RETURN.
E' possibile ritornare indietro per cambiare una riga che non è stata battuta esattamente usando i tasti CRSR e INST/DEL per tornare sull' errore e ribattere. Dopo aver effettuato una modifica o una correzione, premere RETURN per immettere la modifica in quella riga . . . oppure . . . è possibile ribattere qualsiasi riga e qualsiasi commento e premere RETURN per modificarla.

Ecco un altro modo per evidenziare una parola o un titolo, animandoli in modo che sembrino lampeggiare parecchie volte quando compaiono sullo schermo del televisore . . .

40NEXT H

R U N

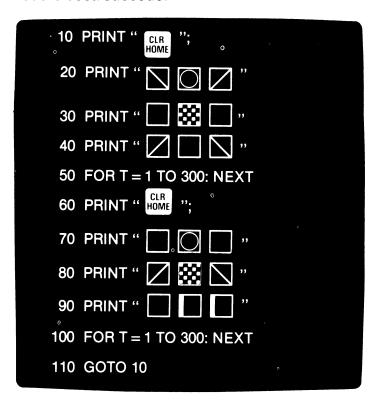
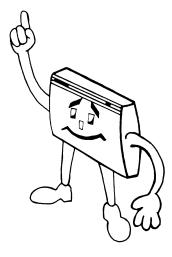
Se il titolo non si sovrappone uniformemente, cercare di regolare la spaziatura all'interno delle virgolette. Per accelerare o rallentare il lampeggio, cambiare il numero nelle righe 25 e 35.

Animazione

- Uccelli volanti
- Pallina rimbalzante
- Controllo del cursore

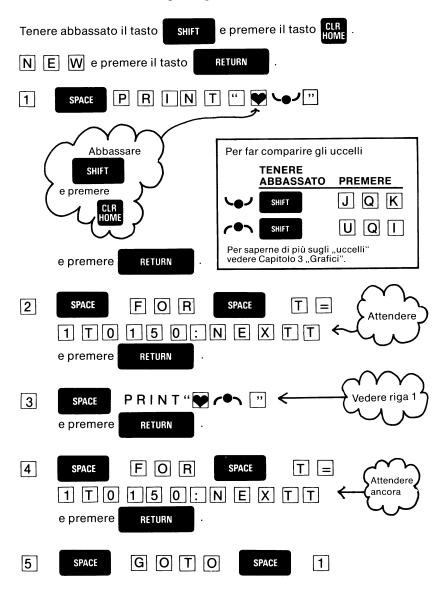
Provare questo programma:

Battere questo programma esattamente come indicato e vedere cosa succede!

```
10 PRINT "[CLR HOME]";
20 PRINT "⧅◯⧄"
30 PRINT "□▒□"
40 PRINT "⧄□⧅"
50 FOR T=1 TO 300: NEXT
60 PRINT "[CLR HOME]";
70 PRINT "□◯□"
80 PRINT "⧄▒⧅"
90 PRINT "□□□"
100 FOR T=1 TO 300: NEXT
110 GOTO 10
```

Per interrompere il programma premere il tasto [RUN STOP].

Uccelli volanti

Questo capitolo spiega come usare le capacità *grafiche* del VIC per creare illusioni di caratteri che si muovono sullo schermo.

L'illusione di movimento che è possibile creare con il VIC è anche detta *animazione*. L'animazione può rendere qualsiasi programma – dai giochi ai programmi gestionali – divertente ed interessante.

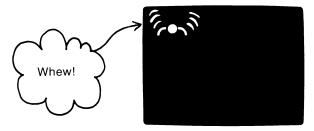
Si inizierà immettendo le righe seguenti:

Cio' che e' stato battuto dovrebbe comparire come segue:

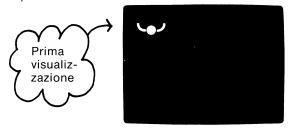
Osservare ciò che è stato battuto. Corrisponde a quanto riportato in figura? In caso contrario correggere le righe inesatte. Quando tutto corrisponde battere:

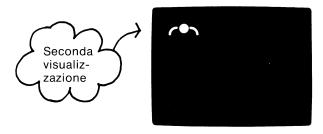
.

SUGGERIMENTO: E' possibile reimmettere o cambiare qualsiasi riga battendola daccapo. La seconda battitura cancellerà automaticamente e sostituirà la prima. E' possibile cancellare una riga battendone soltanto il numero

.

Osservare la prima illusione o animazione col VIC. La creatura «uccello» va saltando e svolazzando nell'angolo superiore sinistro dello schermo. Sembra che usi una notevole potenza delle ali ma deve esserci anche un forte vento di prua. L'uccello non riesce a spostarsi.

E' possibile rendersi conto del perchè sembra che l'uccello muova le ali? Studiare l'esempio per un momento: il segreto di pressochè qualsiasi effetto di animazione sul VIC sta in questo esempio. Come è possibile creare queste illusioni di movimento è facilmente descritto in breve. Ecco un breve sommario di come fare in modo che pressochè qualsiasi carattere si agiti, lampeggi, si muova oscillando, ecc.

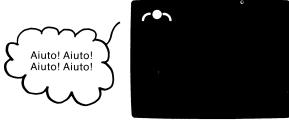
Innanzitutto occorre visualizzare il carattere o la combinazione di caratteri in una posizione sullo schermo.

Secondo, attendere brevemente, servendosi di un'interazione di ritardo del tipo FOR-NEXT. Esempi di istruzioni di ritardo sono indicati nelle righe 2 e 4 del programma degli uccelli volanti. Queste righe dicono al VIC di contare da 1 a 150. Il VIC può contare rapidamente fino a 150. Provare a cambiare 150 in 500 (o in qualche altro numero) e vedere cosa succede.

Terzo, visualizzare il carattere di nuovo ma alterandone qualche parte. A causa del precedente ritardo, l'occhio è ingannato e si pensa di aver visto un «movimento».

Quarto, attendere ancora. Sulla base dell'effetto che si sta cercando di ottenere, questa attesa potrebbe essere più lunga o più breve della prima. Nell'esempio, i ritardi tra le varie comparse sullo schermo erano identici.

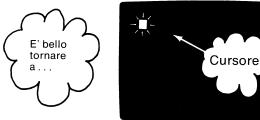
Quinto (la parte facile), *ripetere* le prime quattro fasi appena citate. Ciò si effettua mediante il comando GOTO che dice al VIC di tornare indietro alla prima riga e di ripartire daccapo. In questo modo il programma stampa l'uccello con le sue ali «in alto», conta fino a 150 quindi stampa l'uccello con le sue ali «abbassate» e *torna indietro* ricominciando daccapo.

Nella prima e nella terza fase, dove il carattere viene visualizzato, è spesso necessario assicurarsi che non rimangano sullo schermo vecchie parti del carattere. Nell'esempio dell'uccello, questo problema è risolto inserendo il carattere di cancellazione dello schermo (SHIFT e CLR HOME) in ciascun messaggio. Il carattere *cancella* lo schermo e *riporta il cursore al punto di partenza* in modo che ciascuna versione dell'uccello (con le ali sollevate e con le ali abbassate) compaia nell'angolo superiore sinistro.

E' possibile fare ulteriori esperimenti con questo esempio. Premere per interrompere il battito delle ali e cambiare il limite superiore di conteggio (150) delle righe 2 e 4. Provare a cambiare soltanto un valore lasciando l'altro a 150. L'uccello comincia a librarsi in aria? Provare un valore 50 in entrambi i punti. L'uccello vola più velocemente? Cosa succede con un limite superiore di 500? E di 1000?

Hmmm . . . l'uccello
sembra rallentare

Premere CLR HOME .

Si sa già che è possibile cancellare lo schermo tenendo abbassato SHIFT e premendo il tasto CLR HOME. Dall'interno di un programma è possibile dire al VIC di cancellare lo schermo battendo:

Quando si immette un messaggio di stampa (PRINT) e si premono i due tasti (SHIFT e CLR HOME) che si usano per cancellare lo schermo, nel campo del messaggio compare l'immagine *in negativo* del cuore. Questo simbolo è un segnale per dire al VIC che quando un messaggio viene stampato (PRINT) lo schermo deve essere cancellato in quel punto nel messaggio ed il cursore deve essere rinviato al *punto di partenza.* Il punto di partenza è naturalmente l'angolo superiore sinistro dello schermo.

Provare direttamente l'istruzione PRINT. Batterla nel VIC. Lo schermo si cancella?

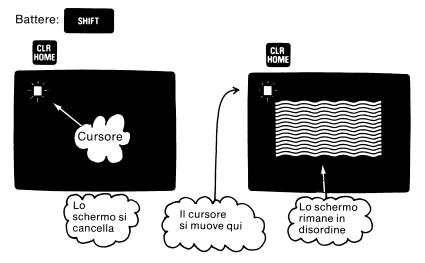
Battere: PRINT « ♥ »;

Ora si tratta di creare disordine sullo schermo. Inserire alcuni uccelli, caratteri, numeri e simboli grafici dappertutto. Quando lo schermo è sufficientemente pieno, battere questa riga:

PRINT « S »;

Notata la differenza con l'altra istruzione PRINT? Sì, questa volta il cursore è tornato al punto di partenza ma lo schermo non si è cancellato.

Se non si preme il tasto SHIFT mentre si batte CLR HOME nel campo del messaggio, compare l'immagine S in negativo invece del cuore. La S in negativo dice al VIC di riportare il cursore in posizione di partenza ma non di cancellare lo schermo.

Battere: SHIFT

Pallina rimbalzante

Cancellare lo schermo ed immettere quanto segue nel VIC:

(Premere RETURN dopo aver battuto ciascuna riga).

NEW
1 PRINT " ● ";

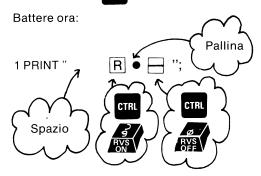
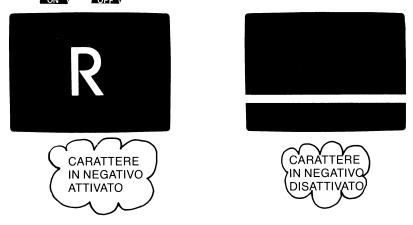
2 GOTO 1

RUN

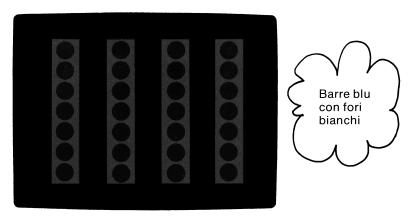
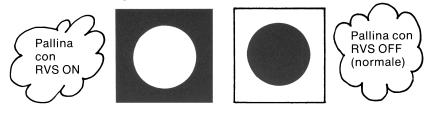
Quando viene battuto RUN, lo schermo dovrebbe riempirsi con colonne di palline blu. Lasciare scorrere le palline per un poco quindi premere

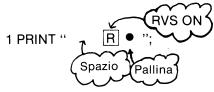
.

La pressione di RUN STOP fa sì che compaiano i messaggi BREAK e READY.

Battere ora:

I due caratteri di immagine in negativo sono i simboli che il VIC inserisce in un campo di messaggio quando si tiene abbassato CTRL e si premono i tasti RVS ON e RVS OFF.

Dopo aver battuto la nuova riga, dire al VIC di eseguire (RUN) questo esempio:

Battere: RUN

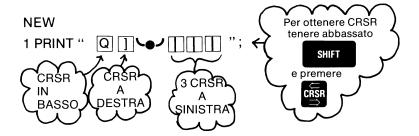
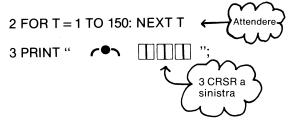
Cosa è successo alle palline? Premere il tasto CTRL per rallentare la stampa (PRINT). Ci sono ora colonne di rettangoli blu con fori bianchi. Perchè?

L'inserimento del carattere di controllo **RVS/ON** davanti al simbolo della pallina fa sì che il VIC *inverta* i colori usati per visualizzare ciascuna pallina. La pallina è normalmente blu ed è circondata da un fondo bianco. Il modo negativo ha reso lo sfondo blu e la pallina bianca.

L'inserimento del carattere di controllo **RVS OFF** dopo la pallina, dice al VIC di ripristinare l'inversione del colore del fondo e dei caratteri. Di invertire cioè il negativo e tornare al punto di partenza. Se non è usato **RVS/OFF**, si produce un effetto interessante.

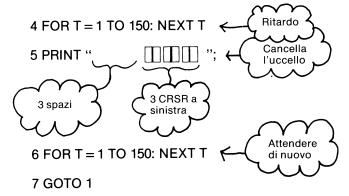
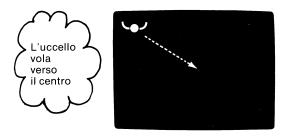
Premere RUN STOP ed immettere questa riga:

Battere: RUN

Quando non viene usato **RVS/OFF**, tutti gli spazi sullo schermo sono resi in negativo e diventano blocchi pieni. Il VIC può trasformarsi in un interessante creatore di profili. Si può addirittura pensare di disegnare i propri tessuti o la carta da pareti o le mattonelle del pavimento con il VIC. Cosa non si può pensare di fare con il VIC?

I tasti del cursore

I quattro tasti di controllo consentono di spostare il cursore ovunque sullo schermo. E' possibile inserire comandi di controllo del cursore nei campi del messaggio delle istruzioni PRINT per facilitare il posizionamento dei caratteri dei messaggi che si stanno stampando. Qui c'è l'esempio degli uccelli dell'inizio del capitolo con l'aggiunta di alcuni comandi del cursore.

Cancellare lo schermo ed immettere una nuova visualizzazione di uccelli:

Questo riga usa parecchi comandi del cursore (verso il basso, verso destra, verso sinistra) per animare e spostare le immagini degli uccelli. Si vedranno i movimenti quando si esegue (RUN) l'intero esempio.

Questa riga contiene tre caratteri di controllo del cursore A SINISTRA.

7 GOTO 1

Battere e controllare l'esempio a fronte del testo stampato. Quando si è pronti, battere RUN ed osservare l'uccellino.

I caratteri di controllo del cursore in questo esempio spostano l'uccello verso il basso in diagonale sullo schermo. Proprio come una gazza!

Ora vale la pena di fare qualche esperimento. Provare ad inventare un'altra creatura e a spostarla attraverso lo schermo.

Animazione con POKE e PEEK

Finora si è imparato ad usare l'istruzione PRINT con i comandi del cursore ed i comandi del colore per spostare gli oggetti sullo schermo. Questo è probabilmente il modo più veloce per spostare gli oggetti usando il BASIC, ma non è il più brillante. La maggior parte dei giochi richiede che l'oggetto si muova all'interno dello schermo e reagisca agli altri oggetti presenti sullo schermo stesso. Il solo modo per programmare ciò è l'uso delle istruzioni PEEK e POKE.

Per usare questa tecnica, occorre per prima cosa conoscere il concetto del video a matrice di punti. Ciascuna posizione dello schermo corrisponde ad una posizione all'interno della memoria del VIC (RAM). Dato che ci sono 506 possibili posizioni sullo schermo per i caratteri, ci sono di consequenza 506 locazioni di memoria destinate a contenere i carattere stessi. E dato che ciascuna locazione di memoria può contenere un numero da 0 a 255, per ciascuna locazione di memoria ci sono 256 possibili valori. Questi sono i 256 diversi caratteri che il VIC può creare (vedere Appendice H). Un numero in una posizione della memoria dello schermo è un codice per il carattere in quella posizione.

La memoria dello schermo nel VIC inizia normalmente alla locazione 7680, e termina alla locazione 8185. La locazione 7680 corrisponde all'angolo superiore sinistro dello schermo. La locazione 7681 è la posizione del successivo carattere alla sua destra e così via lungo la fila. La successiva locazione che segue l'ultimo carattere della fila è il primo carattere della fila successiva più in basso.

Supponiamo ora di voler controllare una pallina che rimbalza sullo schermo. La pallina è al centro dello schermo, alla colonna 10 ed alla fila 10 (l'angolo superiore sinistro è la colonna zero, fila zero). La formula per calcolare la posizione di memoria sullo schermo è:

P = 7680 + X + 22*Y

dove X è la fila e Y è la colonna. Pertanto, la posizione di memoria della pallina 7680 + 10 + 220, ossia 7910. Cancellare lo schermo e battere:

POKE 36879,8

Ciò cambia il colore dello schermo rendendolo completamente nero. Ora battere:

POKE 7910,81

Al centro dello schermo compare una pallina! Si è inserito cioè il carattere direttamente nella memoria dello schermo senza usare l'istruzione PRINT.

La pallina comparsa era bianca ma comunque c'è un modo per cambiare il colore di un oggetto in qualsiasi locazione sullo schermo, anche usando i POKE. Battere:

POKE 38630,3

Il colore della pallina cambia in blu-verde. Per ogni punto sullo schermo del VIC ci sono due locazioni di memoria, una per il codice carattere e l'altra per il codice del colore. La mappa della memoria del colore inizia alla locazione 38400 (angolo superiore sinistro) e continua per 506 locazioni. I codici dei colori da 0 a 7 daranno gli otto colori numerati da 1 a 8 sulla tastiera. (Altri numeri daranno risultati apparentemente strani. Per una spiegazione completa vedere la Guida di Riferimento del Programmatore VIC).

Mappe di memoria dello schermo

CODICI DEI CARATTERI

Questa è la locazione 7681

Questo è il punto 7701

Questa è la locazione di memoria 7680

Questo è il numero 7702

Locazione 7910 (una pallina)

81

Numero 8164

Numero locazione 8185

CODICI DEI COLORI

Questa è la locazione di memoria 38400

Locazione 35421

38630 (blu-verde)

3

Locazione 38884

Locazione 38905

Ecco un breve programma per far rimbalzare una pallina sullo schermo, seguito da una spiegazione dettagliata.

```
 10 PRINT " SHIFT CLR/HOME "
 20 POKE 36879,9
 30 POKE 36878,15
 40 X=1
 50 Y=1
 60 DX=1
 70 DY=1
 80 POKE 7680+X+22*Y, 81
 90 FOR T=1 TO 10: NEXT
100 POKE 7680+X+22*Y, 32
110 X=X+DX
120 IF X=0 OR X=21 THEN DX=-DX: POKE 36876, 220
130 Y=Y+DY
140 IF Y=0 OR Y=22 THEN DY=-DY: POKE 36786, 230
150 POKE 36876, 0
160 GOTO 80
```

La riga 10 cancella lo schermo e la riga 20 rende lo schermo nero con un margine bianco. La riga 30 regola l'intensità del generatore di suoni al livello più elevato.

Le variabili X e Y usate nelle righe 40 e 50 tengono nota della posizione corrente di fila e di colonna della pallina. Le DX e DY usate nelle righe 60 e 70 sono la direzione orizzontale e verticale del movimento della pallina. Aggiungendo un +1 al valore X, la colonna viene spostata di 1 verso destra. Quando viene aggiunto –1 a X, la colonna della pallina viene spostata di 1 posizione verso sinistra. +1 aggiunto a Y sposta la pallina in basso di una fila, e –1 aggiunto a Y sposta la pallina verso l'alto di una fila.

La riga 80 inserisce il carattere della pallina sullo schermo nella sua posizione corrente. La riga 90 è un'interazione di ritardo che lascia la pallina sullo schermo quanto basta perchè l'occhio la possa vedere. La riga 100 cancella ora la pallina inserendo uno spazio (codice di 32) dove si trovava sullo schermo.

La riga 110 aggiunge il fattore di direzione a X. La riga 120 si accerta che la pallina raggiunga una delle pareti laterali, invertendo la direzione ed emettendo un segnale acustico se c'è un rimbalzo. Le righe 130 e 140 fanno lo stesso per le pareti superiore e inferiore.

La riga 150 esclude il suono, eventualmente presente, per concludere l'effetto sonoro del rimbalzo. La riga 160 riporta il programma allo schermo per visualizzare e spostare la pallina di nuovo.

E' bene giocare un poco con questo programma. Cambiando il numero nella riga 80 da 81 ad uno qualsiasi altro codice di carattere, è possibile cambiare la pallina in qualsiasi altro carattere. Se si cambia DX o DY in 0, la pallina rimbalzerà diritta anzichè in diagonale.

E' possibile aggiungere un po' di intelligenza a questo programma. Finora la sola cosa che viene controllata sono i valori di X e Y che rappresentano i confini dello schermo. Aggiungere le righe seguenti al programma sopra indicato. (Basta battere queste righe per aggiungerle automaticamente).

```
 32 FOR L=1 TO 10
 35 POKE 7680 + INT (RND (1) *506), 102
 37 NEXT
155 IF PEEK (7680 + X + 22*Y) = 102 THEN DX = -DX: DY = -DY:
    POKE 36876, 180: GOTO 110
```

Le righe da 32 a 37 inseriscono 10 caratteri grigi sullo schermo in posizioni casuali. La riga 155 controlla («PEEK») che la pallina sia sul punto di rimbalzare in un blocco grigio ed in questo caso ne inverte la direzione.

Vedere l'Appendice H e I per ulteriori informazioni sull'animazione.

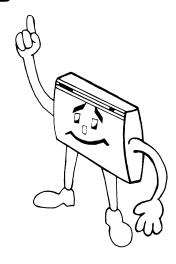
Suono e musica

- Creazione delle note
- Le quattro voci del VIC
- Il generatore di «rumore bianco»
- Esecuzione di motivi
- Uso del VIC come piano
- Alcune parole su POKE

Provare a battere questo programma:

Battere questo programma esattamente come indicato e vedere cosa succede!

```
NEW
 10 PRINT " SHIFT CLR/HOME "
 20 FOR I = 1 TO 5
 30 POKE 36873 + I, 0
 40 NEXT I
 50 PRINT "WHICH VOICE (1-4)?"
 60 PRINT "IF DONE, ENTER 0"
 70 INPUT N
 80 IF N = 0 THEN END
 90 PRINT "WHICH PITCH (128-254)?"
100 INPUT P
110 POKE 36878, 4
120 POKE 36873 + N, P
130 FOR J = 1 TO 2000: NEXT N
140 GOTO 10
```

Per interrompere il programma premere il tasto RUN/STOP.

Creazione delle note

Ci si potrà anche non credere ma col VIC è possibile creare musica ed effetti sonori! Questo capitolo introduce alla musica ed ai suoni che il VIC può generare. Esso insegnerà a produrre questi suoni e ad eseguire qualsiasi tipo di musica da Bach – al rock. Per cui è bene sistemare i candelabri sul televisore e dare inizio al concerto!

Per prima cosa si imparerà a suonare una nota:

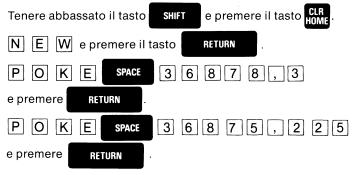
Tenere abbassato il tasto SHIFT e premere il tasto CLR HOME.

N E W e premere il tasto RETURN.

P O K E SPACE 3 6 8 7 8 , 3

e premere RETURN.

P O K E SPACE 3 6 8 7 5 , 2 2 5

e premere RETURN.

A questo punto si dovrebbe sentire un tono proveniente all'altoparlante del televisore. In caso contrario cercare di regolare il volume del televisore nel modo normale, in modo che sia confortevole . . . come la musica di uno show della TV.

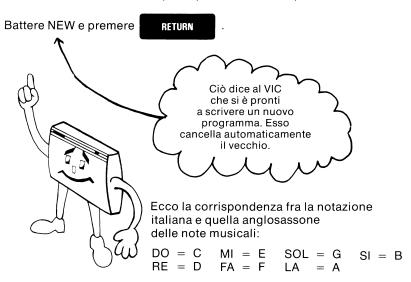
E' stato appena suonato il Do centrale*, una delle 128 note del repertorio di VIC! Per interrompere il tono, battere:

P O K E SPACE 3 6 8 7 5 , 0

Ora possiamo cercare di scoprire quante note il VIC può suonare:

Battere NEW e premere RETURN.

Ecco la corrispondenza fra la notazione italiana e quella anglosassone delle note musicali:

DO = C	MI = E	SOL = G	SI = B
RE = D	FA = F	LA = A	

Battere ora questo programma:

1 POKE 36878, 15 RETURN

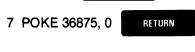
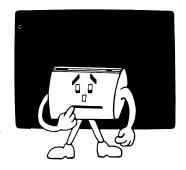
2 FOR I = 128 TO 255 RETURN

3 POKE 36875, I RETURN

4 FOR D = 1 TO 100 RETURN

5 NEXT D RETURN

6 NEXT I RETURN

7 POKE 36875, 0 RETURN

E' opportuno dedicare a questo punto un po' di tempo per osservare lo schermo del VIC e vedere se tutte le righe appaiono esattamente come quelle indicate nel programma. Se qualche riga è sbagliata, basta ribatterla di nuovo. Assicurarsi di comprendere il numero di riga. Il VIC sostituirà automaticamente la nuova riga alla precedente versione. Per ottenere un listato delle righe nel programma battere:

LIST e premere RETURN

Se tutte le righe sono OK, battere:

RUN e premere RETURN

e a questo punto si udranno tutte le 128 note che il VIC può suonare con questa voce. (In caso contrario ricontrollare le righe battute). Se si vogliono ascoltare di nuovo le note, battere:

RUN di nuovo e premere RETURN

Ora che si sono udite le note che il VIC può creare, verrà fornita qualche spiegazione sul modo in cui il VIC procede.

All'interno del VIC, il Video Interface Chip (che dà al VIC il suo nome) si occupa sia del suono che delle immagini e consente al VIC di trasmettere suoni all'altoparlante del televisore. Il volume ed il suono possono essere variati dalla regolazione di volume sul televisore o dalla tastiera del VIC.

Le quattro voci del VIC

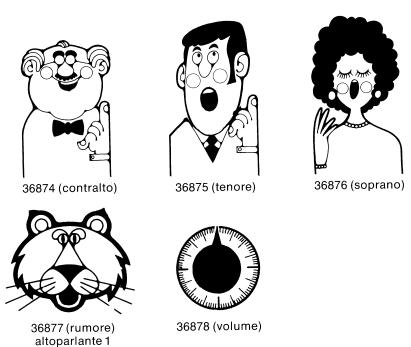
Il VIC ha quattro voci . . . cioè può «cantare» quattro diverse note nello stesso tempo! Si potrebbe pensare a queste voci come quelle di un contralto, di un tenore e ad un rumore. Ciascuna delle voci ha un particolare «numero di controllo altoparlante». Usando questo numero si può attivare l'altoparlante ed usarlo per creare una note musicale o un effetto sonoro. Per far ciò si userà la parola POKE.

I quattro numeri degli altoparlanti del VIC sono:

36874 (altoparlante 1 – musica)
36875 (altoparlante 2 – musica)
36876 (altoparlante 3 – musica) e
36877 (altoparlante 4 – rumore)

Il numero di controllo del volume è 36878.

36874 (contralto) 36875 (tenore) 36876 (soprano)

36877 (rumore) altoparlante 1 36878 (volume)

Per eseguire una nota, battere POKE ed il numero di altoparlante, una virgola ed un numero che rappresenta la nota che si vuole eseguire.

Per esempio, battere:

POKE 36874, 128 e premere **RETURN**

Si è così appena suonato la nota 128 usando l'altoparlante 36874 con la prima voce (la più bassa). D'ora in poi si chiamerà questo altoparlante S1 (abbreviazione per altoparlante 1); il secondo S2; il terzo S3 ed il quarto S4. S2 canta le note più alte di quanto non faccia S1 e S3 canta ancora più alto di S2. S4 canta facendo un rumore . . . creando cioè «rumore bianco», usato per effetti sonori.

Tra l'altro, per escludere una voce, basta eseguire POKE 0 come in questo caso:

POKE 36874,0 e premere RETURN

Ciò potrebbe sembrare un poco complicato per chi non ha pratica di «musica col computer» così abbiamo cercato di facilitare le cose.
E' possibile risparmiare una notevole quantità di lavoro di battitura se si usa il seguente breve programma per tradurre i numeri degli altoparlanti in numeri più corti.

Battere quanto segue sul VIC esattamente indicato:

NEW

S1 = 36874 RETURN

S2 = 36875 RETURN

S3 = 36876 RETURN

S4 = 36877 RETURN

V = 36878 RETURN

Ciò consente di fare riferimento ai numeri degli altoparlanti mediante le abbreviazioni S1, S2, S3, S4 e V (volume). Prima di procedere assicurarsi che ciascuna riga sia stata immessa correttamente.
Ora si è pronti a creare suoni nel modo più facile!

Ciò inserisce (POKE) 10 nella locazione che controlla il volume.
Il controllo di volume può accogliere qualsiasi valore tra 0 e 15. Più alto è il numero e maggiore e più intenso sarà il volume.

```
POKE S1, 195
POKE S2, 215
POKE S3, 231
```

Fare qualche prova con questo metodo per produrre i suoni. E' molto più facile, non è vero? Quando si vuole interrompere il suono (per lasciare riposare le orecchie), eseguire POKE 0 negli altoparlanti che si vogliono spegnere e cioè:

```
POKE S1, 0
POKE S2, 0
POKE S3, 0
```

Ecco una tabella dei valori che è possibile inserire (POKE) negli altoparlanti per ottenere varie note: (**Nota:** I numeri al disotto di 128 producono «silenzio»).

Tabella delle note musicali

(vedere tabella di)
corrispondenza fra notazione italiana e notazione anglosassone a pag. 69)

NOTA	VALORE	NOTA	VALORE
C	135	G	215
C#	143	G#	217
D	147	A	219
D#	151	A#	221
E	159	B	223
F	163	C	225
F#	167	C#	227
G	175	D	228
G#	179	D#	229
A	183	E	231
A#	187	F	232
B	191	F#	233
C	195	G	235
C#	199	G#	236
D	201	A	237
D#	203	A#	238
E	207	B	239
F	209	C	240
F#	212	C#	241

Inserendo (POKE) i valori nelle prime tre voci (S1-S3) è possibile anche suonare dei motivi. («Rumore bianco» non è realmente appropriato per questo tipo di cose . . .

Sfortunatamente questo processo POKE è molto lento e noioso per l'uomo . . . ma è un gioco da ragazzi per il VIC. Cosicché occorre inserire (POKE) in un programma e lasciare che il computer faccia il lavoro! Qualcuno vuole sentire «Flight of the Bumble Bee»? Oppure «Maple Leaf Rag»?

Il generatore di «rumore bianco»

La qúarta voce, S4, è numerata 36877. E' stata chiamata «rumore» in quanto è in effetti un generatore di «rumore bianco», usato per effetti speciali. Si provi ad esempio ad immaginare il ronzìo di un aereo:

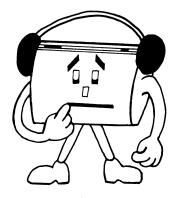
E' un motore a quattro cilindri non è vero? Ora ecco il vento che sbatte tra le ali di un deltaplano.

POKE S4, 240

Non dimenticarsi di aumentare il volume con:

POKE V, 4

o qualsivoglia regolazione di volume si preferisca. Ora escludiamo il suono:

POKE S4, 0

Cercare di sostituire l'istruzione DATA per suonare altri motivi. Per esempio, ecco un vecchio motivo popolare:

```
300 DATA 225, 360, 225, 360, 225, 240
310 DATA 228, 120, 231, 360, 231, 240
320 DATA 228, 120, 231, 240, 232, 120
330 DATA 235, 720, 240, 360, 235, 360
340 DATA 231, 360, 225, 360, 235, 240
350 DATA 232, 120, 231, 240, 228, 120
360 DATA 225, 480
370 DATA -1
```

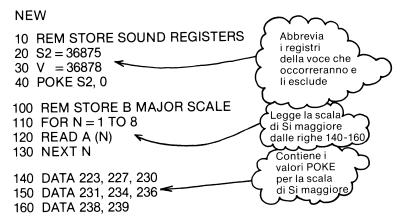
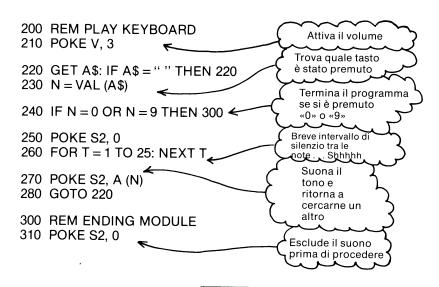
Oppure se si preferiscono i classici:

```
300 DATA 217, 400, 213, 400, 223, 400
310 DATA 227, 200, 234, 200, 230, 400
320 DATA 227, 200, 234, 200, 230, 400
330 DATA 223, 400, 227, 400, 217, 400
340 DATA 213, 600, -1
```

Usare quante righe sono necessarie per suonare motivi più lunghi ma agli effetti di questo programma, mantenersi su tre note per riga

Uso del VIC come piano

Infine, ecco un programma che consente di suonare la tastiera del VIC come un pianoforte:

```
NEW

10 REM STORE SOUND REGISTERS
20 S2 = 36875
30 V  = 36878
40 POKE S2, 0

100 REM STORE B MAJOR SCALE
110 FOR N = 1 TO 8
120 READ A (N)
130 NEXT N

140 DATA 223, 227, 230
150 DATA 231, 234, 236
160 DATA 238, 239
```

Ora, se si batte RUN (e si preme RETURN), è possibile suonare motivi sul VIC. I tasti nella fila superiore (quelli che portano i numeri) controllano le varie note:

****

1	2	3	4	5	6	7	8
DO	RE	MI	FA	SOL	LA	SI	DO

****

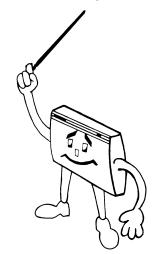
Il VIC continuerà a suonare la nota che si è battuta per ultima fino a che non se ne batte un'altra. Una volta fatto ciò, premere 0 o 9, e la nota cesserà. Per ritornare di nuovo a far sì che il VIC suoni il pianoforte, basta rieseguire (RUN) il programma.

Provare quanto segue (cantando insieme se lo si desidera):

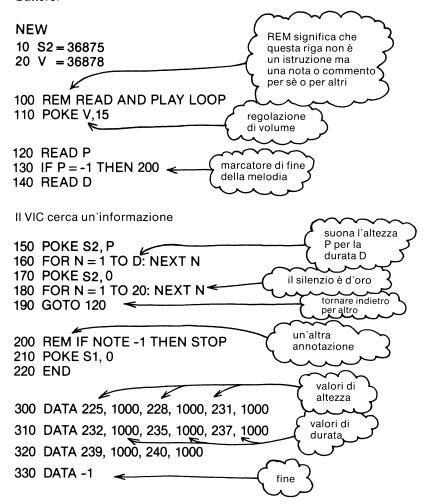
```
1 1 5 5 6 6 5
4 4 3 3 2 2 1
5 5 4 4 3 3 2
5 5 4 4 3 3 2
1 1 5 5 6 6 5
4 4 3 3 2 2 1 8
9
```

OPPURE:

```
3 3 4 5 5 4 3 2
1 1 2 3 3 2 2
3 3 4 5 5 4 3 2
1 1 2 3 2 1 1
0
```

Prendi e porta a casa, Ludwig!

Esecuzione di motivi

Usando gli altoparlanti del VIC e la tabella dei valori delle note musicali, è possibile creare proprie canzoni trascrivere motivi da un libro di musica. Il seguente programma indica come procedere:

Battere:

Nel primo gruppo di istruzioni abbiamo impostato le locazioni POKE per l'altoparlante o gli altoparlanti che stiamo usando, in questo caso l'altoparlante 2 ed il volume ed abbiamo immesso le nostre abbreviazioni. Il successivo gruppo di istruzioni inizia alla riga 100.

La riga 100 contiene una nota (REM) che spiega cosa si pensa che questa sezione debba fare. E' detta iterazione o «loop» in quanto la sezione leggerà e suonerà una nota quindi ritornerà all'inizio e rifarà lo stesso per un'altra nota. La riga 110 attiva il volume.

Ora diremo al VIC di trovare la nota da suonare:

Suggerimento VIC:

I programmi non devono iniziare con la riga 1 o essere numerati per 1. La maggior parte dei programmi inizia a 10 e procede ad incrementi di 10. In questo modo è possibile aggiungere righe in più nel mezzo, se lo si desidera. Per esempio, se potrebbe aggiungere le righe 11, 12, ecc. tra le righe 10 e 20.

La riga 120 dice al VIC cercare nel programma e leggere (READ) le informazioni – chiamiamole P – sulla nota da suonare. Questa informazione è contenuta in una «istruzione misteriosa» che non abbiamo ancora scritto. Analogamente, la riga 140 dice al VIC di leggere (READ) informazioni – chiamiamole D – sulla durata della nota.

Notare particolarmente la riga 130. La funzione di questa riga è di interrompere il programma quando è stata letta l'ultima nota. Senza qualche tipo di marcatore di «fine melodia» il programma cercherà di leggere note che non sono ancora state scritte e compirebbe un errore. La riga 130 dice che quando il VIC legge questo marcatore – un valore di –1 – non deve cercare di suonare questa nota ma di portarsi su un modulo di fine alla riga 200. Occorre ricordarsi di inserire pertanto –1 al termine della nostra «istruzione misteriosa».

Ora che siamo in grado di far suonare al VIC una nota, interrompiamola e cerchiamone un'altra.

La riga 150 semplicemente inserisce (POKE) la nota che abbiamo letto (READ) nella riga 120 nella voce 1 mentre la riga 160 crea un ritardo della durata che abbiamo letto (READ) nella riga 130. Analogamente, le righe 170 e 180 escludono la voce 1 per un breve periodo. La riga 190 riporta il VIC alla riga 120 per leggere (READ) la successiva nota.

Ora occorre scrivere la sezione finale. Ricordarsi che il marcatore di «fine della melodia», –1, rimanda il programma alla riga 200.

La riga 210 esclude la voce e la riga 220 dice al VIC di interrompere l'esecuzione delle istruzioni di questo programma.

Quantunque sia stata scritta una sezione finale, non abbiamo ancora finito. Dobbiamo infatti scrivere le nostre «istruzioni misteriose» per dire al programma quali note leggere (READ). Queste istruzioni misteriose sono dette istruzioni DATA in quanto contengono informazioni o dati. Le istruzioni DATA possono essere disposte ovunque in un programma. Ogniqualvolta il VIC incontra un'istruzione READ, cerca un'istruzione DATA da leggere (READ).

Questo modulo contiene i DATA per una scala in Do maggiore. La riga 300 contiene le prime tre note. Il primo numero è il valore POKE per le prima nota, 225 – il Do basso. Il secondo numero ne definisce la durata, 1000 – circa 1 secondo. La riga 310 contiene le successive tre note e la riga 320 le ultime due. I valori sono presi dalla tabella delle note musicali che precede.

Se le righe sembrano corrette, è possibile iniziare l'esecuzione (RUN). Si dovrebbe in questo caso udire una scala di Do maggiore pressochè precisa! Se ci sono una stonatura o due, provare a regolare i valori nell'istruzione DATA iniziando dalla riga 300.

Ancora una volta: E' possibile inserire istruzioni DATA ovunque nel programma. Esse verranno lette (READ) una per una, iniziando con la riga di numero inferiore e passando attraverso ciascuna istruzione DATA dall'inizio alla fine.

Alcune parole su POKE

Il comando POKE consente di trattare con il VIC su un livello completamente nuovo. POKE consente di trovare una particolare locazione di memoria e cambiare ciò che vi è memorizzato. Dato che questo comando opera direttamente sulla memoria del VIC. è possibile compiere errori inserendo (POKE) valori, nelle locazioni sbagliate o valori sbagliati nelle posizioni giuste! Vogliamo a questo proposito ripetere ciò che è stato detto nel Capitolo 1: *Non c'è modo di danneggiare il computer battendo sulla tastiera . . .* neppure con POKE. Ma è possibile far sì che il VIC vada in qualche altro posto e metta il broncio, interrompendo qualsiasi contatto. Per esempio, se si è pronti a terminare questa lezione in ogni caso, provare a battere:

POKE 788,0

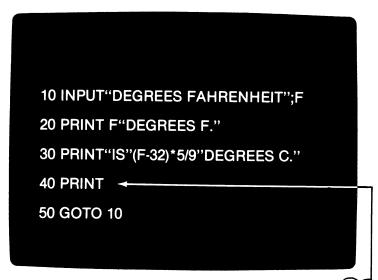
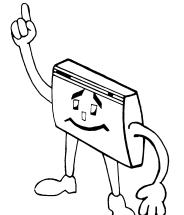
Si può trovare che il solo modo in cui è possibile riguadagnare una comunicazione con un computer che è stato «insultato» in questo modo consiste nel battere RUN/STOP e RESTORE. Se il «guasto» è serio, è possibile dover spegnere e riaccendere il computer. Questo non lo danneggia ma significa che qualsivoglia programma su cui si stia lavorando vada perso. Anche se questo dovesse succedere, il danno si limita ad un lavoro supplementare di battitura. Ma suggerisce che bisogna fare attenzione a ciò che si inserisce (POKE). Ciò sottolinea anche l'importanza degli accessori tipo le unità disco e le cassette di dati che possono memorizzare programmi nonchè le stampanti che possono almeno dare un listato di programma che può aiutare a ricostruirne uno perso.

Si consiglia di effettuare un breve studio di POKE nell'Appendice C prima di mettersi a «POKE» quà e là indiscriminatamente nel VIC. Provare almeno gli esempi forniti nel manuale. Fare attenzione a battere quei lunghi numeri e controllare a fondo il lavoro prima di eseguire un programma. Un computer in cui è stato eseguito un POKE nel posto sbagliato può offendersi e ritirarsi seccato in un angolo.

In questo capitolo si è imparato a suonare Bach e Beethoven col VIC. Allora, maestro, prego!

La conversazione col VIC

- Qual è il proprio nome
- Presentazione delle variabili
- Scegliere una nota
- L'istruzione GET

Provare a battere questo programma:

Battere questo programma esattamente come indicato e vedere cosa succede!

La battitura della parola PRINT di per sè su una riga aggiunge una riga vuota quando il programma viene eseguito. Provare con e senza questa riga.

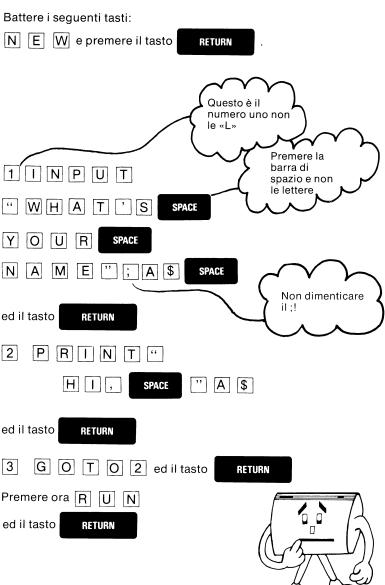
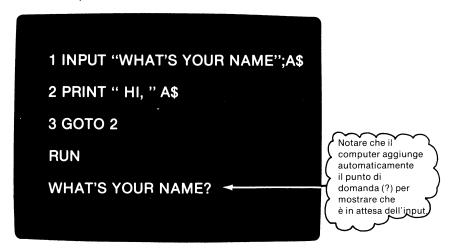
Per interrompere il programma, premere il tasto **RUN STOP** e premere **RESTORE** .

Buongiorno VIC. La tua missione (noi sappiamo che tu l'accetterai) è di eseguire miracoli ogniqualvolta noi tocchiamo pochi tasti!

Sì il VIC è pronto e ben disposto a rispondere alle varie richieste più di qualsiasi altro computer mai finora costruito. Allacciare dunque le cinture di sicurezza e prepararsi ad un divertimento senza fine!

Tenere abbassato il tasto SHIFT e premere il tasto CLR HOME.

Battere i seguenti tasti:

N E W e premere il tasto RETURN.

Ed il VIC risponde:

A questo punto si batte la risposta abituale (il proprio nome) (se il nome fosse VIC occorrerebbe battere:

E voilà! Il nome è comparso sullo schermo! Per rallentare lo show basta premere il tasto CTRL . Quando si è pronti a rinunciare alla la celebrità, premere il tasto RUN STOP .

Per immettere un altro nome battere RUN per vedere ricomparire la stessa *richiesta.* La richiesta in questo caso è: WHAT'S YOUR NAME? (QUAL E' IL TUO NOME?)

Le richieste sono domande dirette all'operatore nel tentativo che il VIC fa per ottenere informazioni. Il VIC ha un numero limitato di modi per acquisire informazioni dal mondo dell'uomo. Probabilmente il metodo più utilizzato dal computer per raccogliere informazioni dalla tastiera è l'istruzione INPUT. Esaminiamo ora le varie fasi del programma. Ecco cosa abbiamo detto al VIC di fare:

Innanzitutto visualizzare il messaggio «WHAT'S YOUR NAME» sullo schermo e quindi attendere l'immissione di caratteri dalla tastiera (INPUT). Prendere la risposta e denominarla «A$». Nell'esempio «VIC» diventa A$. Questo è un tipo di stenografia che usa il computer.

Secondo, stampare la parola «HI» seguita da qualsiasi cosa venga battuto sulla tastiera. Nel nostro esempio abbiamo stampato HI VIC.

Terzo, ritornare alla riga numero 2.

La seconda e la terza fase si uniscono per far sì che il VIC stampi in continuazione il messaggio «HI» su tutto lo schermo. Se si cambiasse la riga 3 per farla comparire come:

3 GOTO 1

il messaggio HI VIC si stamperebbe sullo schermo soltanto una volta sola e comparirebbe di nuovo la richiesta «WHAT'S YOUR NAME». Questa modifica fa sembrare che il VIC abbia l'amnesia!

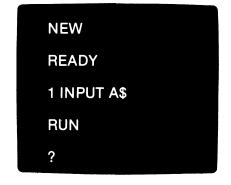
Ogniqualvolta si usa l'istruzione INPUT, il programma blocca tutto in attesa di una risposta. E' importante notare che il VIC attende indefinitamente o fino a che non viene premuto il tasto RETURN, quello che viene prima.

Nel programma abbiamo creato una richiesta amichevole. A meno che non si dica al VIC cosa vogliamo che il nostro messaggio di input dica, otterremo un semplice «?» che non ci dice molto. Cosicchè cercheremo di costruire richieste che suggeriscono quale tipo di input è richiesto.

Premere RUN STOP e RESTORE contemporaneamente.

Battere NEW e

1 INPUT A$

Quando si batte RUN lo schermo del VIC appare come segue:

Il punto di domanda da solo pone più problemi di quanti non ne risolva! Proviamo in questo modo:

1 PRINT "MAY I HAVE YOUR NAME":INPUT A$

2 PRINT "WHAT IS YOUR FAVORITE FOOD":INPUT B$

3 PRINT

4 PRINT "THANK YOU,"A$" FOR YOUR POLITE ANSWER"

5 PRINT "WE WILL GIVE YOU SOME SPACE "B$" SPACE AS A GESTURE OF OUR APPRECIATION"

Dopo aver battuto RUN e RETURN si vedrà che trattare i messaggi con rispetto può tradursi in una minaccia ai propri sforzi!

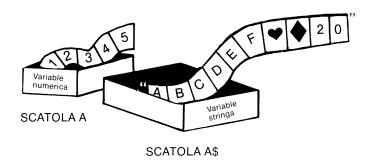
Presentazione delle variabili

Molti dei programmi usati in questo manuale usano variabili per semplificare o rafforzare il programma.

Le variabili sono molto utili in quanto possono essere usate per rappresentare numeri, formule, simboli grafici, parole, frasi intere. Esempi di nomi variabili sono: X, AB, S2, X$, AB$, S2$. Il modo più semplice per spiegare la potenza di una variabile è di dire che queste semplici variabili possono essere usate ciascuna per rappresentare fino a 255 caratteri!

Ci sono due tipi di variabili: variabili numeriche e variabili stringa. Le variabili numeriche sono usate per memorizzare numeri (in effetti valori numerici). Le variabili stringa possono essere usate per memorizzare tutti i tipi di caratteri (numeri, lettere, grafici, comandi del cursore, controlli del colore, ecc.).

SCATOLA A

SCATOLA A$

Le variabili sono come contenitori posti all'interno del computer. Per dire al VIC se un contenitore è per i numeri o i valori, occorre usare un nome speciale. I nomi delle variabili numeriche possono essere lunghi uno o due caratteri e possono essere costituiti da una lettera, due lettere o da una lettera e da un numero. Ecco alcuni esempi di nomi di variabili numeriche:

X AB S2 C2 AA ZX

Le variabili stringa possono essere lunghe uno, due o tre caratteri (compreso il segno $), devono sempre iniziare con una lettera da A a Z e terminare con il segno del dollaro ($). Ecco alcuni esempi di variabili stringa:

X$ AB$ S2$ C2$ AA$ ZX$

Ecco un breve programma che mostra un modo per usare le variabili:

```
10 A$ = "VIC 20"

20 PRINT"HELLO,"A$

RUN
```

Il VIC visualizzerà HELLO, VIC 20. Perchè? Perchè nella riga 10 si è detto al VIC che la variabile A$ equivale a «VIC 20».
Provare ora con questo:

```
10A = 2

20B = 3

30C = 4

40PRINT A*B*C
```

In questo esempio, A*B*C è identico a 2*3*4 in quanto sono stati memorizzati i numeri 2, 3 e 4 nelle variabili A, B e C.

Ecco un esempio finale che usa due tipi di variabili con le istruzioni INPUT. L'istruzione INPUT consente alla persona che esegue il programma di definire che cosa rappresenta la variabile come in questo caso:

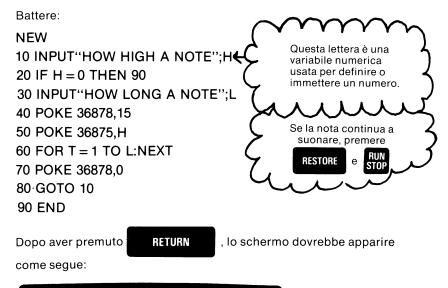
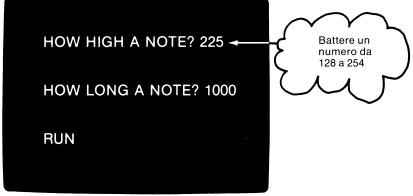
```
10PRINT"WHAT WORD DO YOU WANT X$ TO STAND FOR":
  INPUT X$

20PRINT"WHAT NUMBER DO YOU WANT X TO STAND FOR":
  INPUT X

30PRINT"NOW X$ STANDS FOR" X$

40PRINT"AND X STANDS FOR" X

RUN
```

Scegliere una nota

Per questo esempio INPUT, ci divertiremo un po' con il suono del VIC.

Battere:

```
NEW
10 INPUT"HOW HIGH A NOTE";H
20 IF H=0 THEN 90
30 INPUT"HOW LONG A NOTE";L
40 POKE 36878,15
50 POKE 36875,H
60 FOR T=1 TO L:NEXT
70 POKE 36878,0
80 GOTO 10
90 END
```


Dopo aver premuto RETURN, lo schermo dovrebbe apparire come segue:

Premere il tasto RETURN ed ascoltare l'altezza e la durata della nota. Dopo ciascuna nota è possibile crearne un'altra ed avere un'idea di come questi «misteriosi» POKE si traducono in suoni.

Le prime due righe di «Creazione di una nota» (10 e 20) danno la flessibilità necessaria per variare l'altezza rispettivamente nella riga 50 e 60. La riga 30 è la nostra «porta» per uscire quando abbiamo finito di sperimentare (come battere 0 in risposta a HOW HIGH A NOTE? – quanto è alta una nota? La riga 70 spegne la voce che era stata attivata precedentemente nella riga 40.

Se si vogliono conoscere tutti i dettagli importanti per creare la musica con il VIC, passare al capitolo cinque.

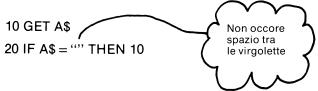
L'istruzione GET

Ora che si è in grado di padroneggiare l'istruzione INPUT, ci dedicheremo ad un modo più vantaggioso per ottenere informazioni dalla tastiera.

L'istruzione GET viene usata per trasferire caratteri dalla tastiera, un carattere alla volta. Infatti, la persona che esegue (RUN) il programma non deve nemmeno battere il tasto RETURN ! Ecco come si presenta l'istruziono:

```
10 GET A$
```

Come creare un arresto di programma ed attendere che venga battuto qualcosa? Basta inserire il programma in un loop o iterazione con un'istruzione IF . . . THEN che controlla una risposta.

```
10 GET A$
20 IF A$ = "" THEN 10
```

Qual è l'uso di ciò? Per un'applicazione molto semplice, questo piccolo programma in 2 righe consente al programma di fare una pausa fino a che l'operatore non batte un tasto sulla tastiera. Ciò aiuta a «congelare» lo schermo fino a che la persona non l'ha letto e desidera procedere.

Ecco una spiegazione ampliata per l'istruzione GET:

```
10 GET A$
20 IF A$ = "" THEN 10
30 IF A$ = "A" THEN PRINT "CHICKEN SOUP"
40 IF A$ = "B" THEN PRINT "SPAGHETTI"
50 IF A$ = "C" THEN PRINT "STEAK AND EGGS"
60 GOTO 10
RUN
```

Una volta che questo programma è battuto ed eseguito (RUN), il VIC attenderà che l'operatore batta qualsiasi tasto. Se il tasto è la lettera A, sullo schermo compaiono le parole CHICKEN SOUP. La lettera B fa comparire la parola SPAGHETTI e la parola C fa comparire le parole STEAK AND EGGS. Ora abbiamo la possibilità di far battere al VIC parole intere con una sola operazione sui tasti!

Ora esamineremo il programma, il più lungo finora visto, un esempio pratico delle istruzioni GET e PRINT usate per creare una *raccolta di ricette* computerizzata. Non bisogna però allarmarsi dalle dimensioni di

questo programma. Esso usa prevalentemente semplici istruzioni PRINT ma la lezione da imparare qui è il modo in cui la variabile A$ è usata per sostituire parecchie intere frasi e come il comando GET viene usato.

```
10 PRINT " [SHIFT][CLR HOME] PLEASE PICK A CHOICE"
20 PRINT "FROM THE MENU:"
30 PRINT
40 PRINT "A...CHICKEN SOUP"
50 PRINT "B...SPAGHETTI"
60 PRINT "C...STEAK & EGGS"
200 GET A$:IFA$ = ""THEN 200
210 IF A$ = "A" THEN 500
220 IF A$ = "B" THEN 700
230 IF A$ = "C" THEN 900
490 GOTO 200
500 PRINT " [SHIFT][CLR HOME] MIKE'S CHICKEN SOUP"
510 PRINT
520 PRINT "TAKE 1 CHICKEN. KILL"
```

```
530 PRINT "AND PLUCK. REMOVE"
540 PRINT "GIBLETS. BOIL 4 QTS"
550 PRINT "WATER IN A LARGE POT."
560 PRINT "ADD CHICKEN. BOIL"
570 PRINT "2 HOURS, OR UNTIL"
580 PRINT "HOUSE SMELLS GOOD."
590 PRINT
```

Provare a battere proprie ricette! Tutto ciò che si deve fare è di cambiare i titoli e le informazioni della ricetta . . .

Ciò significa che se si batte qualsiasi altra cosa non succede nulla ed il programma attende che venga battuto A, B o C

Notare che parti delle frasi sono stampate su righe separate per rendere più facile la lettura

La battitura di PRINT su una riga a sè inserisce uno «SPAZIO» di una riga sullo schermo

```
600 PRINT "HIT ANY KEY TO GO ON"
610 GET A$:IF A$ = "" THEN 610
620 GOTO 10
700 PRINT " [SHIFT] [CLR HOME] MA'S SPAGHETTI"
710 PRINT
720 PRINT "BROWN 1 LB. GROUND"
730 PRINT "BEEF, WITH 1 ONION"
740 PRINT "AND 1 GREEN PEPPER."
750 PRINT "ADD 1 LG. CAN TOMATO"
760 PRINT "PUREE, 6 OZ. CAN TOM."
770 PRINT "PASTE, 6 OZ. WATER,"
780 PRINT "3 CLOVES GARLIC, SALT"
790 PRINT "& PEPPER, RED PEPPER,"
800 PRINT "OREGANO. SIMMER 1 HR"
810 PRINT "& SERVE WITH COOKED"
820 PRINT "NOODLES."
830 GOTO 590
900 PRINT " [SHIFT] [CLR HOME] STEAK AND EGGS"
910 PRINT
920 PRINT "TAKE 1 COOKED STEAK"
930 PRINT "AND COOKED EGGS."
940 PRINT "SERVE TOGETHER WITH"
950 PRINT "BEVERAGE."
960 GOTO 590
```

Se si è battuto il programma correttamente e battuto RUN, lo schermo dovrebbe comparire con la seguente visualizzazione:

```
PLEASE PICK A CHOICE
FROM THE MENU:

A...CHICKEN SOUP
B...SPAGHETTI
C...STEAK & EGGS
```

Ora il VIC è in attesa che venga battuto un tasto. Se si batte qualsiasi altra cosa diversa da A, B o C, non succede nulla (se ne occupa la riga 200). Se si batte A, significa che si cerca la ricetta della Chicken Soup alla Mike. Interessante vero?

Ciò può essere allungato e modificato molto facilmente a secondo dell'uso che se ne vuole fare. Per aggiungere varie voci al menù, basta aggiungere un'istruzione PRINT dopo la riga 60, aggiungere una nuova istruzione IF dopo la riga 230 ed aggiungere la ricetta ovunque c'è spazio al termine. L'ultima riga della ricetta dovrebbe essere la riga GOTO 590 che dice alla persona che esegue (RUN) il programma di battere un tasto per continuare. Ciò mantiene inoltre la ricetta sullo schermo fino a che non la si è completamente eseguita.

E' possibile usare il programma appena descritto per qualcosa di diverso dalle ricette, naturalmente. Cosa dire di un archivio di nomi e indirizzi? Invece di un «menù» usare cognomi con iniziali. Invece delle ricette usare il nome della persona, l'indirizzo ed il numero di telefono. E' possibile pensare ad altri usi per programmi basati su GET e INPUT? Questa è la vera potenza di calcolo – la possibilità cioè di adattare alle proprie esigenze ciò che il computer può fare.

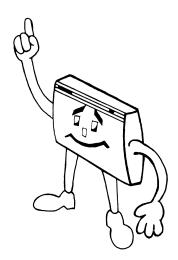
Introduzione alla programmazione

- I primi programmi BASIC
- Numeri casuali

Provare a battere questo programma:

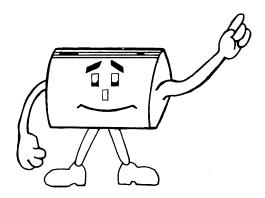
Battere questo programma esattamente come indicato e vedere cosa succede!

Per interrompere il programma, premere il tasto

.

I primi programmi BASIC

Finora, sono stati battuti parecchi programmi senza però comprenderne il funzionamento. Questo capitolo spiegherà come sono composti i piccoli, furbi programmi e consentirà di procedere ulteriormente verso la programmazione del proprio VIC.

PROGRAMMA 1: Il proprio nome illuminato
(Capitolo 2)

Non sembra esserci molto su questo piccolo programma ma una volta che si comprende cosa sta facendo, c'è la chiave per eseguire effetti di animazione. La riga 10 è il comando PRINT, con il carattere che significa «cancellare lo schermo» all'interno delle virgolette. Se si cercasse di battere la riga senza le virgolette, la riga di programma scomparirebbe dallo schermo prima ancora di finire di batterla. *Il VIC riconosce soltanto una nuova riga quando si batte il tasto RETURN con il cursore su quella riga.*

Suggerimento VIC: Virgolette

parleremo ora un poco su quanto succede quando si batte il tasto delle virgolette. La prima volta che si batte il tasto delle virgolette, accade qualcosa di divertente. Se si batte HOME, CLR, cursore verso l'alto, cursore verso il basso, verso sinistra e verso destra si ottiene un *carattere grafico ed in negativo.* Ciò succede anche sul VIC quando viene premuto uno qualsiasi dei tasti di controllo del colore. Come si vede, i segni di virgolette sono usati nei programmi di computer ed il VIC riconosce questi segni come un comando di programmazione. Pertanto quando si batte un tasto di controllo del cursore o del colore dopo una serie di virgolette, il VIC visualizza un codice speciale per individuare quell'operazione. Per esempio quando si batte " e si preme SHIFT CLR HOME, si ottiene un cuore in negativo sullo schermo. Se si vede comparire questo comando in un programma, si a che significa «cancellare lo schermo». Altri simboli corrispondono ad altre operazioni. Una volta che si batte il tasto delle virgolette per la *seconda volta* qualsiasi tasto di controllo del cursore e di colore funzionerà normalmente.

Il tasto INSERT provoca un effetto simile. Quando viene battuto questo tasto, ogni spazio creato fra le righe agisce come se fosse nel modo «virgolette». Tutti i caratteri di controllo del cursore compariranno come se si trovassero all'interno delle virgolette. Inoltre, il tasto DELETE produrrà un carattere grafico speciale in negativo in questi spazi, che avrà l'effetto di cancellare caratteri quando il programma viene listato e di stampare DELETE sullo schermo quando si esegue la stampa (PRINT).

La riga 20 del programme è detta un'*iterazione di ritardo.* Si tratta in effetti di due istruzioni BASIC su una riga, separate l'una dall'altra dal due punti (:). Tutti ciò che succede qui è che il video conterà da 1 a 300, senza fare altro. Ciò serve a rallentare leggermente il programma (provare a cancellare le righe 20 e 40 e ad eseguire – RUN – il programma. Esso lampeggia troppo velocemente!)

Nella riga 30, viene battuto il nome all'interno delle virgolette (almeno si spera che lo si sia fatto). Ciò ha fatto sì che il nome venisse stampato sullo schermo. Non c'è altro sullo schermo in quanto la riga 10 lo aveva già cancellato.

La riga 40 é un altro ritardo per dare al nome il tempo di rimanere sullo schermo quanto basta poterlo leggere.

La riga 50 fa sì che il programma «salti» alla riga 10 come riga successiva da eseguire e cancella brevemente lo schermo.

Una volta che il nome è comparso sullo schermo della TV, c'è un ritardo di un secondo dopo di che lo schermo viene cancellato. Dopo un altro secondo, il nome viene visualizzato di nuovo nella stessa posizione. Di nuovo viene cancellato e così via. Perchè le lettere compaiono nella stessa posizione dello schermo, l'occhio crede che le lettere lampeggino.

Provare ora a fare qualche esperimento con questo programma. E' possibile allungare o ridurre gli intervalli tra la visualizzazione e la cancellazione del nome cambiando il numero 300 nelle righe 20 e 40.

PROGRAMMA 2: Molti cuori
(Capitolo 3)

```
10 FOR H = 1 TO 505
20 PRINT " ♥ ";
30 NEXT
40 FOR C = 8 TO 255 STEP 17
50 POKE 36879, C
60 FOR T = 1 TO 500: NEXT
70 NEXT
80 GOTO 40
```

Questo programma crea uno schermo pieno di cuori colorati. Esso introduce l'uso dei segni di punteggiatura nelle istruzioni PRINT e l'uso di POKE per cambiare il colore dello schermo e dei margini.

La riga 10 imposta un'iterazione che conta da 1 a 505. Si vuole cioè che compaiano 505 cuori sullo schermo in quanto sullo schermo ci sono 506 spazi. Se si stampasse (PRINT) il 506-esimo carattere, lo schermo verrebbe costretto a scorrere verso l'alto di una riga e ci sarebbero *meno* cuori sullo schermo di prima.

La riga 20 stampa (PRINT) il carattere del cuore sullo schermo. Il punto e virgola (;) dopo l'ultimo segno di virgolette ha un effetto importante. Come si sa, dopo una normale istruzione PRINT, il VIC esegue automaticamente due operazioni – sposta cioè il cursore all'indietro all'inizio della riga (e cioè esegue un *ritorno a capo)* e sposta il cursore verso il basso alla riga successiva (esegue cioè un'*interlinea).*
Il segno di punteggiatura al termine della riga cancella il ritorno a capo e l'interlinea in modo che la successiva cosa che viene stampata (PRINT) compare direttamente alla destra dell'ultima cosa stampata (PRINT).

La riga 30 completa semplicemente l'iterazione di ritardo. Dato che il valore di H è 505 o meno, il programma stamperà i cuori sullo schermo. Quando viene stampato il 505-esimo cuore, il programma continua con la riga successiva (riga 40).

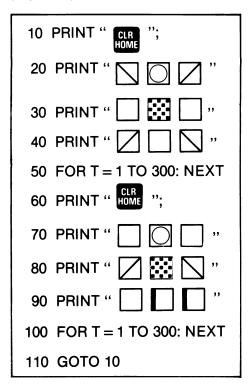
La riga 40 crea una nuova iterazione e definisce la riga 50, che cambia i colori del margine e dello schermo. C è definito come una serie di numeri da 8 a 255, che aumenta a incrementi di 17. Ogni volta che viene battuta la riga 70 (NEXT), viene aggiunto 17 al valore precedente di C e la somma viene usata per il nuovo valore. Ciò fa sì che venga scelto un ciclo di colori che comprende il margine nero con lo schermo nero, il margine bianco con lo schermo bianco, ecc. per tutti gli otto colori del margine. Quindi i numeri in C vanno oltre i valori di quei colori e scelgono 7 colori diversi per lo schermo. (Vedere Appendice I per una lista dei numeri dei colori).

La riga 50 è l'istruzione che effettivamente cambia il colore. Il valore contenuto nella variabile C è memorizzato nella posizione di memoria 36879. Si tratta in effetti di una locazione sullo stesso chip VIC e non nella normale area di memoria.

La riga 60 è un'iterazione di ritardo. Se questa riga venisse rimossa, i colori cambierebbero in maniera così veloce da far venire il mal di testa.

La riga 70 completa l'iterazione iniziata nella riga 40. Notare che la riga si sarebbe potuta scrivere nella forma 70 NEXT C, ma non deve esserlo.

La riga 80 fa riprendere al programma il ciclo attraverso i vari colori. Esso continuerà all'infinito a meno che non si prema il tasto STOP o non si spenga il VIC. Battendo POKE 36879, 27 dopo aver battuto STOP, si ripristinano i normali colori dello schermo.

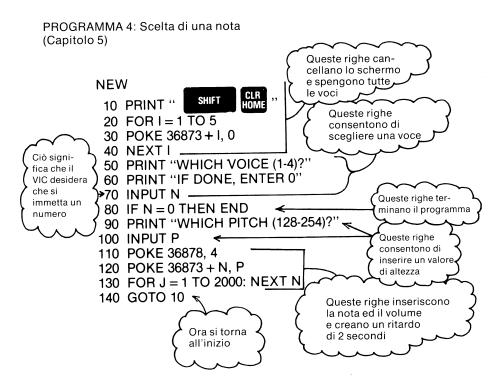
PROGRAMMA 3: Esercizio con il signor VIC
(Capitolo 4)

Questo programma è simile al primo eseguito, quello che presentava sullo un nome illuminato lampeggiante, ma in questo caso invece di disegnare un'immagine e quindi farla lampeggiare, come faceva il primo programma, disegna un'immagine completa, fa una pausa, sostituisce l'immagine con un'altra immagine completa. Mentre la testa ed il corpo del signor VIC rimangono nella stessa posizione, le braccia e le gambe cambiano posto. Ciò dà l'illusione del movimento da una posizione alla successiva.

Le righe 10 e 60 portano il cursore all'angolo superiore sinistro dello schermo, note come *posizione di partenza.* Ciò fa sì che l'immagine del signor VIC venga visualizzata ogni volta nella stessa posizione sullo schermo.

Le righe 20, 30 e 40 «disegneranno» ciascun tratto della prima immagine del signor VIC.

Le righe 50 e 100 sono iterazioni di ritardo e servono a mantenere l'immagine sullo schermo per un tempo sufficiente.

Le righe 70, 80 e 90 disegnano la seconda immagine sullo schermo. Ciò avviene in maniera così veloce da non rendere possibile osservare la transizione – l'occhio vede un passaggio istantaneo da un'immagine all'altra.

PROGRAMMA 4: Scelta di una nota
(Capitolo 5)

Una volta verificato che tutte le righe siano corrette, provare a eseguire questo programma (battere RUN e premere RETURN). Ciò consente di scegliere una voce ed un valore di altezza e di suonare il tono scelto per circa due secondi. Il suono si interrompe ed il programma chiede un'altra voce ed un altro valore di altezza. Il programma è la delizia di chi fa esperimenti musicali. Quando si vuole interrompere il programma, immettere 0 come scelta per la voce. E' possibile pensare che c'è un problema quando si sceglie l'altezza 254 nella voce 3 e non si ode nulla. In effetti non si tratta di un errore – questa nota è semplicemente troppo alta per le orecchie dell'uomo. (Si potrebbe però provarla sul cane!)

PROGRAMMA 5: Conversione di temperatura
(Capitolo 6)

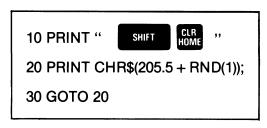
```
10 INPUT "DEGREES FAHRENHEIT"; F

20 PRINT F "DEGREES F."

30 PRINT "IS" (F-32)*5/9 "DEGREES C."

40 PRINT

50 GOTO 10
```

Qui viene introdotta l'istruzione INPUT, che consente al programma di interrompere ciò che sta facendo e richiedere le necessarie informazioni all'operatore (la persona cioè che esegue – RUN – il programma).

La riga 10 fa sì che sullo schermo compaia il messaggio DEGREES FAHRENHEIT? (gradi Fahrenheit?). Le parole all'interno delle virgolette di un'istruzione INPUT funzionano esattamente come l'istruzione PRINT. In ogni caso, l'ultima parola sarà sempre seguita da un carattere punto interrogativo (?) ed il programma attenderà a questo punto ulteriori informazioni.

La riga 20 stampa i valori di F, che e ciò che è stato battuto (immesso). La riga 30 stampa il risultato del calcolo di conversione. Nella riga 40 la parola PRINT da sola su una riga fa sì che sullo schermo compaia una riga vuota.

Infine la riga 50 fa sì che il programma torni all'inizio e richieda altre informazioni per ripartire. Verrà posta di nuovo la domanda originale e si potranno convertire altre temperature. Al termine, tenere abbassato il tasto STOP e battere il tasto RESTORE. Non c'è alcun modo per dire al programma che si è finito.

PROGRAMMA 6: Labirinto casuale
(Capitolo 7)

Questo è un piccolo e semplice programma che stampa sullo schermo pseudo-labirinti. Come si può immaginare, la riga importante è la numero 20.

La funzione CHR$ dà un carattere basato su un numero di codice da 0 a 255. Ogni carattere che il VIC può inserire sullo schermo è codificato in questo modo (vedere Appendice H). Per trovare il codice di ciascun carattere, basta battere PRINT ASC(«X») dove X è il carattere che si

sta controllando. Quindi battere PRINT CHR$(X) dove X è il numero che il VIC ha fornito. Visto come funziona?

Ora provare a battere PRINT CHR$ (205); CHR$ (206). Ciò dovrebbe stampare i due carattere grafici posti sul lato destro sui tasti M e N. Questi sono i due caratteri che il programma usa per costruire i «labirinti».

Aggiungendo la formula 205.5 + RND(1), il VIC sceglie un numero casuale tra 205.5 e 206.5. Ci sono pari probabilità che un numero risulti al disopra o al disotto di 206. La funzione CHR$, ignora qualsiasi valore frazionario. Pertanto per metà delle volte verrà stampato il carattere con il codice 205 e per l'altra metà il carattere con il codice 206.

Se si vuole fare esperimenti con questo programma, provare ad aggiungere o a sottrarre un paio di decimi a 205,5. Ciò dà all'uno o all'altro carattere una maggior possibilità di comparire.

Altro sui numeri casuali

La funzione «numero casuale» è uno degli aspetti più utili e più gradevoli del BASIC, che consente di programmare tutti i tipi di giochi di probabilità.

La riga X=RND(1) fa sì che il VIC scelga un numero casuale tra 0 e 1 esclusi, da inserire in X. Ciò si traduce in un campo di possibili valori per X:

$0 < X < 1$

Quando si lavora con i numeri casuali, vale la pena di tenere presente che si genera un campo di numeri per vedere come i calcoli interessano l'intero campo. Per esempio se si volesse ottenere una serie di possibili valori tra 0 e 3, si potrebbe semplicemente moltiplicare X per tre. Il nuovo campo è:

$0 < X < 3$

Se occorresse scegliere un numero da 10 a 20, come sarebbe possibile eseguire un calcolo per cambiare il campo. Innanzitutto occorrerebbe aggiungere 10 al numero scelto, per cambiare il campo in

$10 < X < 11$

Moltiplicando il numero casuale per 10 prima di aggiungere il 10, il campo diventa:

$10 < X < 20$

Cosicchè la formula per un numero casuale tra 10 e 20 è:

X=RND(1)*10 + 10

Finora abbiamo imparato come cambiare il campo dei possibili risultati per il numero casuale. In ogni caso il risultato della funzione conterrà ingombranti cifre decimali che non sono desiderabili per applicazioni tipo lancio di dadi o la scelta di un numero da 1 a 10. La funzione usata per ripulire tutto questo ciarpame è INT, che elimina tutte le cifre decimali dal numero. La formula per un numero casuale da 10 a 20, con l'aggiunta della funzione INT diventa pertanto: X=INT(RND(1)*10 + 10).

Il campo di possibili risultati è ora:

10 < = X < = 19

Ma attenzione! Il limite superiore del campo è sceso da 20 a 19 in questo caso. Perchè? Perchè prima il campo era sempre minore di 20. La funzione INT toglie le cifre decimali da un numero maggiore di 19 e minore di 20 per dare luogo a 19. All'altra estremità della gamma, qualsiasi risultato tra 10 e 11 viene troncato a 10. Se occorresse invece avere un campo di numeri da 10 a 20, la formula dovrebbe diventare:

X=INT(RND(1)*11 + 10)

Il numero casuale viene moltiplicato per espandere il campo ed aggiunto per spostarlo.

La formula generale per una serie di numeri casuali in un certo campo è:

X=INT(RND(1)*a) + b

Dove a rappresenta il numero di possibili risultati e b è il numero minore nel campo.

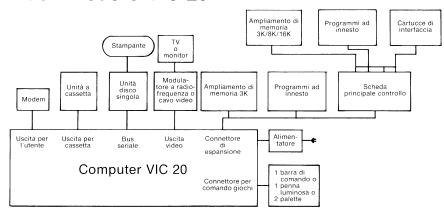
Appendici

Appendice A: Gli accessori VIC – una rapida Introduzione

Questo è un manuale per i principianti cosicchè non si dedicherà molto tempo e spazio a parlare delle varie periferiche che si inseriscono nelle varie uscite nella parte posteriore del VIC.

Ciascuna delle periferiche avrà un proprio manuale di istruzioni che spiegherà come collegarla al VIC ed usarla per fare cose interessanti.

La COMMODORE ha disegnato il VIC per crescere insieme alle esigenze dell'utente e prevede un piano globale relativo al modo in cui le varie periferiche si uniscono fra di loro e col VIC. Ecco una mappa di tale piano.

Il Sistema di Personal Computer Commodore VIC 20*

Programmi speciali ad innesto

Super Expander Cartridge
- Memoria aggiuntiva 3K (porta il VIC a 8K)
- Grafici ad alta risolvenza e comandi tracciamento
- Tasti di funzione preassegnati

Programming Aid Cartridge
- La «cassetta di attrezzi» del programmatore Machines-Code Monitor
- Accessoal linguaggio macchina
- Tasti di funzione preassegnati (comandi di programma)
- Tasti di funzione assegnabili dall'utente

Registratore a cassetta

La prima periferica che probabilmente si acquisterà sarà il registratore a nastro COMMODORE. Il registratore può accogliere parecchie migliaia di caratteri (lettere e numeri) su una normale cassetta magnetica. E' possibile memorizzare programmi abbastanza lunghi su nastri e quindi caricarli abbastanza rapidamente, senza doverli ribattere ogni volta (vedere Appendice B - Lavoro con le cassette magnetiche).

Il registratore a cassetta VIC si inserisce ad innesto nell'uscita di interfaccia per cassetta sul dorso del VIC con la sua spina speciale e non richiede qualsiasi particolare interfaccia. La COMMODORE fornisce una varietà di programmi di computer su nastro per l'impiego con il registratore a cassetta.

Pannello di espansione multiplo

E' possibile inserire un pannello di comando principale nel connettore di espansione sul dorso del VIC. Questa unità di controllo consente di usare più di una cartuccia alla volta. Esso prevede sei uscite ed accetta cartucce di programma, ampliamenti di memoria e comprende un'interfaccia IEEE-488 che consente di usare le periferiche PET/CBM Commodore ed anche unità IEEE.

Unita' Floppy Disk singola VIC

L'unità disco floppy singola VIC può memorizzare fino a 170 000 caratteri (lettere e numeri) e memorizzare un programma molto lungo caricandolo o togliendolo dalla memoria del VIC in una frazione di secondo. Non occorre quindi aspettare come nel caso dell'unità nastro che il programma venga trovato, letto e caricato. Questo dispositivo si collega al connettore seriale del VIC.

Cartucce di interfaccia IEEE-488

IEEE-488 è uno standard universale scientifico che consente di usare le periferiche COMMODORE PET/CBM tipo unità dischi e stampanti, nonchè strumenti ed attrezzi scientifici.

Stampante seriale

Il VIC ha un'interfaccia seriale per le periferiche che usano collegamenti seriali per comunicare con il computer. Con la speciale stampante a matrice a punti del COMMODORE che usa questo collegamento seriale è possibile stampare su carta i listati di programma ed i risultati.

Connettore per comando giochi

Il connettore per comando giochi del VIC consente di collegare barre di comando, penne luminose e palette in modo da far funzionare facilmente gli stessi giochi delle sale specializzate senza dover usare la tastiera.

Telecomunicazioni

Si sarà già letto di programmi che consentono al personal computer di parlare ad altri computer e di accedere ai servizi informazioni specializzati che si stanno creando un poco ovunque. Nel VIC questa capacità è già incorporata.

L'uscita per l'utente del VIC comprende un'interfaccia R5232 che consente di ricevere o scambiare informazioni sulla linea telefonica usando un *modem* poco costoso per telecomunicazioni.

Software

Oltre a tutto questo hardware, il VIC presenta un assortimento di software interessante.

Alcuni programmi sono memorizzati in cartucce che si inseriscono ad innesto nella parte posteriore del VIC e sono pronte per il funzionamento non appena l'apparecchio viene acceso. Alcuni programmi sono su nastro, altri su disco e molti sono compresi nel VIC Learning Series Book e nella serie di cartucce che consentono di imparare il calcolo ed altre materie.

Ma questa è stata semplicemente una rapida panoramica degli accessori disponibili.

TENERSI A CONTATTO CON IL
RIVENDITORE COMMODORE PER CONOSCERE
LE INFORMAZIONI PIÙ RECENTI
RELATIVE AI NUOVI SVILUPPI!

Appendice B: Uso del registratore a cassetta

Il registratore a cassetta è la prima periferica che viene acquistata dalla maggior parte dei possessori del VIC. Si tratta di un dispositivo che consente di registrare i programmi per successivo impiego ed inoltre di memorizzare e richiamare dati. Questa Appendice sarà di aiuto una volta che si disporrà del registratore a cassetta.

Il primo comando che si userà con il registratore è SAVE. Quando viene immesso questo comando, il VIC risponde con il messaggio PRESS PLAY & RECORD ON TAPE (PREMERE PLAY E RECORD). Occorre cioè tenere abbassato il tasto RECORD sul registratore e quindi premere PLAY. Entrambi i tasti dovrebbero rimanere abbassati mentre il VIC sta registrando il programma (se il tasto RECORD non resta abbassato, controllare il dorso della cassetta per accertarsi che la piccola linguetta non sia stata staccata. Se è stata staccata, il VIC non consente di registrare su quel nastro. Questa linguetta deve essere usata per un nastro sul quale non si può scrivere). Una volta che è abbassato il tasto PLAY, il VIC dà il messaggio OK e fa comparire sullo schermo la parola SAVING (salvataggio in corso). Dopo un breve periodo, compare il messaggio OK e la parola READY (Pronto) e l'operazione di salvataggio (SAVE) è completata. E' possibile dare un nome ad un programma inserendolo tra virgolette dopo la parola SAVE (vedere Appendice C).

Una volta che il programma è stato salvato (SAVE), è una buona idea controllarlo per assicurarsi che la versione sul nastro sia OK. E' possibile infatti che la registrazione non sia riuscita, ad esempio per la presenza di un difetto sul nastro. A questo scopo il VIC prevede un comando noto come VERIFY, che controlla il programma sul nastro a fronte della copia del programma esistente nella memoria RAM.

Quando si batte la parola VERIFY, il VIC visualizza il messaggio PRESS PLAY ON TAPE (premere PLAY). Se si fa seguire la parola VERIFY da un nome di programma, sul nastro verrà esaminato quel programma altrimenti verrà verificato (VERIFY) il primo programma sul nastro. Quando si preme il tasto PLAY, il video risponde con OK e con la parola SEARCHING (Ricerca in corso). Quindi il VIC elenca qualsiasi programma con nomi diversi da quello che si sta cercando, esattamente per far sapere che essi sono presenti. Quando è stato individuato il nome del programma cercato, compare il messaggio FOUND e VERIFYING (Trovato e Verifica in corso). Il VIC ora confronta il programma su nastro con quello che c'è in memoria, procedendo fino alla fine anche se trova un errore.

Quando viene raggiunta la fine del programma sul nastro, viene visualizzato il messaggio OK e READY (OK e Pronto) se tutto è a posto oppure il messaggio ?VERIFY ERROR (Verificare errore) se c'è un problema con il programma così come compare sul nastro.
Se si incontra un errore, provare ad usare un nastro vuoto diverso e ripetere l'operazione.

Quando si vuole caricare un programma da nastro nella memoria interna del VIC, usare il comando LOAD. Il VIC dice PRESS PLAY ON TAPE ed inizia a caricare il programma. Al termine corretto dell'operazione, si riceve il messaggio READY. Se c'è un problema con il programma sul nastro, compare sullo schermo un messaggio ?LOAD ERROR (Errore di caricamento). Se il nastro è *molto* difettoso, possono succedere cose stranissime al VIC. Potrebbero comparire caratteri divertenti sullo schermo o potrebbe succedere qualsiasi altra cosa. In caso di caricamento difettoso come questo, probabilmente la miglior cosa è di spegnere il VIC e riaccenderlo e riprovare.

Ora veniamo ad un aspetto diverso dell'uso della cassetta di nastro e cioè la memorizzazione di dati (ossia variabili e risultati dei calcoli) sul nastro ed il loro richiamo.

Per iniziare a memorizzare dati su nastro, la prima operazione da fare è di creare il file su nastro. Ciò avviene usando l'istruzione OPEN, che è seguita da tre numeri e da un nome come questo:

OPEN 1,1,1,«HELLO»

Il primo numero che segue la parola OPEN è il numero del file per mezzo del quale si farà riferimento successivamente al file nel programma. Il secondo numero è un numero di unità, che sarà 1 per la cassetta. Il terzo numero è 0 per la lettura, 1 per la scrittura e 2 per la scrittura con marcatore di fine nastro che segue il file. Il nome HELLO sarà inserito su nastro per identificare il file consentendo in questo modo l'uso sul nastro di molti file con nomi diversi. Notare che durante la scrittura non viene cercato uno spazio vuoto sul nastro e durante la lettura del nastro il VIC cerca il file con un dato nome. L'istruzione OPEN sopra indicata creerà un file su nastro per la scrittura che verrà chiamato HELLO e cui si farà riferimento come al file N. 1 da parte delle successive istruzioni nel programma.

Per scrivere effettivamente i dati sul nastro, viene usata l'istruzione PRINT#. Il segno # è seguito dal numero del file (nel suddetto esempio un 1) quindi da una virgola e quindi dall'elenco delle voci che vengono scritte sul nastro. Questa lista dovrà essere dello stesso formato usato quando si stampa (PRINT) sullo schermo. Una virgola provoca spazi extra tra le voci il che rappresenta uno spreco di spazio sul nastro dato che gli spazi vuoti occupano altrettanto spazio dei caratteri. Normalmente su questa riga viene usato il punto e virgola (;) come separatore.

Si noterà scrivendo i file su nastro che la cassetta non gira in continuazione ma parte e si ferma ripetutamente. Ciò in quanto il VIC sta *bufferizzando* i dati (ossia li scrive su una memoria di transito). Ciò significa che i dati non vengono scritti direttamente su nastro ma vengono disposti in un'area speciale della memoria RAM detta buffer o memoria di transito, area che può contenere fino a 192 caratteri. Quando nel buffer viene inserito il 192-esimo carattere, il VIC cessa di inviare caratteri al buffer e copia l'intero buffer sul nastro come un blocco unico.

Quando si è terminato di scrivere i dati sul nastro, occorre chiudere (CLOSE) il file usando l'istruzione CLOSE. La parola CLOSE è seguita dal numero del file che si vuole chiudere.

Ciò è necessario a causa della presenza del buffer. Se ci fossero ancora caratteri nel buffer quando si è finita la scrittura, questi caratteri andrebbero persi. L'operazione CLOSE memorizzerà tutti i caratteri rimanenti nel buffer ed apporrà sul nastro un marcatore di fine file (o fine nastro).

Per leggere i dati da nastro, occorre per prima cosa aprire (OPEN) il file come indicato sopra usando il numero 0 come terzo numero che segue la parola OPEN. Il VIC cercherà il nome del file, se specificato ed il nastro si interromperà una volta che l'ha trovato. Ora verranno usate le istruzioni INPUT# o GET# per richiamare i dati. Se il buffer è vuoto quando viene eseguito INPUT# o GET#, verrà caricato da nastro il successivo buffer pieno. I dati vengono cioè letti dal buffer e non direttamente dal nastro.

La differenza tra INPUT# e GET# è che INPUT# prende un'intera variabile scritta su nastro, numero o stringa, mentre GET# prende soltanto un carattere alla volta. In effetti, se si prova con GET# si scoprirà che il carattere dopo l'ultimo di ciascun elemento di dati è un CHR$ (13) che sta per ritorno a capo. Questo carattere viene usato per separare gli elementi di dati su nastro, che è il modo per far sapere a INPUT# dove si trova la fine di una variabile.

Qui ci sono tre semplici programmi per consentire di fare esperimenti con la lettura e la scrittura di file su nastro. Il primo programma creerà il file. Il secondo deve essere immesso dopo che il primo è stato eseguito (RUN) ed il nastro di dati è stato riavvolto e dimostra l'uso di INPUT# per richiamare informazioni. Il terzo programma usa l'istruzione GET# per richiamare i dati carattere per carattere e stampa non solo il carattere ma anche il suo codice ASCII.

Programma 1: Scrittura su nastro

10 PRINT" SHIFT CLR HOME WRITE-TO-TAPE PROGRAM"

20 OPEN 1,1,1,"DATA FILE"

30 PRINT"NOW TYPE DATA TO BE"

40 PRINT"STORED OR TYPE   STOP

50 PRINT

60 INPUT"DATA";A$

70 PRINT#1,A$

80 IFA$<>"STOP"THEN50

90 PRINT

100 PRINT"CLOSING FILE"

110 CLOSE 1

Programma 2: Lettura del nastro usando INPUT#

10 PRINT"   READ-TAPE PROGRAM"

20 OPEN 1,1,0,"DATA FILE"

30 PRINT"FILE OPEN"

40 PRINT

50 INPUT#1,A$

60 PRINT A$

70 IF A$="STOP"THEN END

80 GOTO 40

Programma 3: Lettura del nastro usando GET#

Le righe da 10 a 40 sono identiche al PROGRAMMA 2

50 GET#1, A$

60 IF A$=""THEN END

70 PRINT A$,ASC(A$)

80 GOTO 50

Appendice C: il BASIC del VIC

Questo manuale ha fornito un'introduzione al linguaggio BASIC, quanto basta per avere un'idea della programmazione di computer e di una parte del vocabolario coinvolto. Questa Appendice fornisce un elenco completo delle regole (SINTASSI) del linguaggio BASIC VIC unitamente ad una descrizione concisa di ciascuna di esse. Occorre naturalmente far pratica con questi comandi ricordando che non è possibile provocare alcun danno permanente al VIC semplicemente battendo programmi e che il modo migliore per imparare il calcolo è di eseguirlo.

Questa Appendice è divisa in sezioni secondo i diversi tipi di operazioni nel BASIC. Queste comprendono:

1. **Variabili ed operatori:** descrive i diversi tipi di variabili, i nomi variabili leciti e gli operatori logici ed aritmetici.
2. **Comandi:** descrive i comandi usati per lavorare con i programmi, correggerli, memorizzarli e cancellarli.
3. **Istruzioni:** descrive le istruzioni di programma BASIC usate nelle righe numerate dei programmi.
4. **Funzioni:** descrive le funzioni stringa, numeriche e di stampa.

I comandi in ciascuna sezione sono indicati alfabeticamente per comodità. Nella guida di riferimento del programmatore VIC è fornita una spiegazione più completa dei comandi del BASIC VIC; la Guida viene fornita insieme al VIC.

1. Variabili e operatori
a. Variabili

Il VIC usa tre tipi di variabili in BASIC. Queste sono: **numeriche normali, numeriche intere e stringa (alfanumeriche).**

Le variabili numeriche normali dette anche variabili in virgola mobile, possono avere qualsiasi valore da -10^{38} a $+10^{38}$, con una precisione massima di nove cifre. Quando un numero supera le nove cifre, ad esempio 10^{10} o 10^{-10}, il computer lo visualizza in notazione scientifica usando il numero normalizzato ad una cifra e otto cifre decimali seguite dalla lettera E e dalla potenza di dieci per la quale il numero è moltiplicato. Per esempio, il numero 12345678901 verrà visualizzato nella forma 1.23456789E +11.

Le variabili intere sono usate quando il numero sarà sempre compreso tra +32767 e –32768 e senza parti frazionarie.

Le variabili intere richiedono meno spazio di memoria delle variabili in virgola mobile ma la differenza probabilmente non sarebbe sostanziale a meno che usate in grandi quantità tipo in una matrice (vedere più avanti). Una variabile intera potrebbe essere rappresentata da un numero tipo 5, 10 o –100.

Le variabili stringa sono quelle usate per i dati alfanumerici e possono contenere numeri, lettere e qualsiasi altro carattere che il VIC può creare. Un esempio di variabile stringa è «VIC20».

I nomi variabili possono consistere in una singola lettera, in una lettera seguita da un numero o in due lettere.

Una variabile intera è specificata usando il segno di *percento (%)* dopo il nome variabile. Le variabili stringa sono seguita dal *segno del dollaro* $.

ESEMPI: Nomi variabili numeriche: A, A5, BZ
Nomi variabili intere: A%, A5%, BZ%
Nomi variabili stringa: A$, A5$, BZ$

Le *matrici* sono liste di variabili con lo stesso nome che usano un numero extra per specificare la loro natura. Esse sono definite usando l'istruzione DIM e possono contenere variabili in virgola mobile, intere o stringa. Il nome variabile stringa è seguito da una serie di parentesi () che racchiude il numero della variabile nella lista.

ESEMPI: A(7),BZ%(11),A$(87)

Le matrici possono avere più di una dimensione. Una matrice bidimensionale può essere pensata come costituita da file e da colonne, con il primo numero nella parentesi che indica la fila ed il secondo numero che indica la colonna.

ESEMPI: A(7,2),BZ%(2,3,4),Z$(3,2)

Ci sono tre nomi variabili riservati al VIC che non possono essere usati per uno scopo normale. Si tratta delle variabili ST, TI e TI$. ST è una variabile di stato che si riferisce alle operazioni di input/output. Il valore di ST cambia se c'è un problema nel caricamento di un programma o di dati dal nastro o dal disco. Una spiegazione più dettagliata di ST si trova nel Manuale di Riferimento del Programmatore VIC BASIC.

TI e TI$ sono variabili che si riferiscono all'orologio in tempo reale incorporato nel VIC. La variabile TI è aggiornata ogni sessantesimo di secondo. Essa inizia a 0 quando il VIC viene acceso e viene ripristinata soltanto cambiando il valore di TI$.

TI$ è una stringa che è costantemente aggiornata dal sistema. I primi due caratteri contengono il numero delle ore, il terzo ed il quarto carattere sono i numeri dei minuti ed il quinto e sesto carattere sono i numeri dei secondi. A questa variabile può essere attribuito qualsiasi valore (dato che tutti i caratteri sono numeri) e verrà aggiornata automaticamente da quel punto.

ESEMPIO: TI$=«101530» predispone l'orologio a 10:15 e 30 secondi (anti-meridiane).

Questo orologio viene cancellato quando il VIC viene spento, e riparte da zero quando il VIC viene di nuovo riacceso.

b. Operatori

Gli operatori aritmetici comprendono i segni seguenti:

+ addizione
– sottrazione
* moltiplicazione
/ divisione
↑ elevamento a potenza

Se una riga contiene più di un operatore, c'è un *ordine* ben definito in cui le operazioni si devono sempre verificare. Se vengono usati insieme parecchi operatori, il computer assegna le priorità come segue: innanzitutto l'elevamento ad esponente, successivamente la moltiplicazione e la divisione ed infine l'addizione e la sottrazione. Se si vuole che queste operazioni si verifichino in un ordine diverso il VIC consente di isolare un calcolo circondandolo da parentesi. Le operazioni racchiuse tra parentesi avvengono prima delle altre. Assicurarsi che le formule abbiano lo stesso numero di parentesi di apertura e di chiusura altrimenti il programma dà luogo ad un messaggio SYNTAX ERROR quando viene eseguito.

Ci sono inoltre operatori per uguaglianze e disuguaglianze:

=	uguale a
<	minore di
>	maggiore di
< = oppure = <	minore di o uguale a
> = oppure = >	maggiore di o uguale a
<> oppure ><	diverso da

Infine ci sono tre operatori logici:

AND OR NOT

Questi sono usati più spesso per riunire formule multiple nelle istruzioni IF . . . THEN.

ESEMPIO:

IF A=B AND C=D THEN 100	richiede che A=B e C=D siano veri
IF A=B OR C=D THEN 100	consente che A=B oppure C=D siano veri

2. Comandi

CONT (Continua)

Questo comando viene usato per riprendere l'esecuzione di un programma che era stato interrotto usando il tasto STOP, l'istruzione STOP o un'istruzione END all'interno del programma. Il programma ripartirà nel punto esatto in cui si era fermato.

CONT non funziona se sono state modificate o aggiunte righe al programma (o se si è semplicemente spostato il cursore ad una riga di programma e quindi si è battuto RETURN senza cambiare nulla) o se il programma si è fermato a causa di un errore oppure se si è provocato un errore prima di cercare di far ripartire il programma. Il messaggio in questo caso è CAN'T CONTINUE ERROR (Errore, non posso continuare).

LIST

Il comando LIST consente di osservare le righe di un programma BASIC che sono state battute o caricate (LOAD) nella memoria del VIC. Quando usato da solo senza essere seguito da numeri fa comparire sullo schermo una lista completa del programma (che può essere rallentata tenendo abbassato il tasto CTRL o interrotta battendo il tasto contrassegnato RUN STOP). Se si fa seguire la parola LIST da un numero riga, il VIC mostrerà soltanto quel numero di riga. Se si batte LIST con due numeri separati da un trattino, il VIC mostrerà tutte le righe tra il primo ed il secondo numero. Se si batte LIST seguito da un numero e da un solo trattino, il VIC presenterà tutte le righe da quel numero fino alla fine del programma. E se si batte LIST, un trattino e quindi un numero, si otterranno tutte le righe dall'inizio fino a quel numero di riga. Usando queste variazioni è possibile esaminare qualsiasi parte del programma o portarne le righe sullo schermo per la modifica.

ESEMPI:

LIST	Mostra l'intero programma
LIST 10–	Mostra soltanto dalla riga 10 fino al termine
LIST 10	Mostra soltanto la riga 10
LIST–10	Mostra la riga dall'inizio fino alla riga 10
LIST 10–20	Mostra le righe da 10 a 20 comprese

LOAD

Questo è il comando da usare quando c'è un programma memorizzato sulla cassetta o sul disco e lo si vuole usare. Se si batte semplicemente LOAD e il tasto RETURN, il VIC troverà il primo programma sulla cassetta e lo porterà in memoria per essere eseguito (RUN), listato (LIST) o altro. E' possibile inoltre battere la parola LOAD seguita da un nome di programma, il che deve essere un nome tra virgolette («»). Il nome può essere seguito da una virgola (al di fuori delle virgolette) e da un numero o variabile numerica che agisce come numero di dispositivo per determinare l'origine del programma. Se non viene indicato alcun numero, il VIC presume il dispositivo N. 1 che è il registratore a cassetta.

L'altro dispositivo comunemente usato con il comando LOAD è l'unità disco, che è il dispositivo N. 8.

ESEMPI:

LOAD	Legge il successivo programma su nastro
LOAD «HELLO»	Cerca nel nastro il programma denominato HELLO e lo carica se lo trova
LOAD A$	Cerca un programma il cui nome è nella variabile denominata A$
LOAD «HELLO»,8	Cerca il programma denominato HELLO sull'unità disco
LOAD«*»,8	Cerca il primo programma sul disco

Il comando LOAD può essere usato all'interno di un programma BASIC per trovare ed eseguire (RUN) il successivo programma sul nastro.

NEW

Questo comando cancella l'intero programma in memoria e cancella inoltre le variabili eventualmente usate. A meno che il programma non sia precedentemente memorizzato altrove, va perso fino a che non lo si batte di nuovo. Fare quindi attenzione usando questo comando.

Il comando NEW può anche essere usato come un'istruzione in un programma BASIC. Quando il VIC raggiunge questa riga, il programma viene cancellato e tutto si ferma. E' utile se si vuole lasciare tutto pulito quando il programma è finito.

RUN

Quando un programma è stato battutto nella memoria o caricato (LOAD), il comando RUN viene usato per farlo funzionare. Se il comando RUN non è seguito da un numero, il computer inizierà con la riga di programma di numero minore. Se c'è un numero, questo indica il numero di riga dal quale inizia il programma.

ESEMPI:

RUN	Inizia l'esecuzione del programma partendo dal numero di riga minore
RUN 100	Inizia il programma alla riga 100
RUN X	SYNTAX ERROR! (Occorre sempre battere RUN da solo o con un numero di riga – non con una lettera).

SAVE

Questo comando memorizza su una cassetta o su disco un programma correntemente in memoria. Se si batte la parola SAVE e si preme RETURN, il VIC tenterà di memorizzare il programma sulla cassetta.

Esso non ha alcun modo per controllare che c'è già un programma in quel punto cosicchè bisogna fare attenzione con i nastri. Se si batte il comando SAVE seguito da un nome tra virgolette o da un nome variabile stringa, il VIC attribuirà al programma quel nome cosicchè potrà essere facilmente individuato e richiamato in futuro. Il nome sarà seguito da una virgola (dopo le virgolette) e da un numero o variabile numerica. Questo numero dice al VIC qual è il dispositivo sul quale memorizzare il programma. Il dispositivo numero 1 è l'unità nastro e quello numero 8 l'unità disco. Dopo il numero può esserci una virgola ed il secondo numero può essere 0 o 1. Se il secondo numero è 1, il VIC sceglierà un marcatore END-OF-TAPE (fine nastro) dopo il programma. Se si cerca di caricare (RUN) un programma ed il VIC trova uno di questi marcatori, si ottiene un messaggio FILE NOT FOUND ERROR (Errore, file non trovato).

ESEMPIO:

SAVE	Memorizza il programma su nastro senza un nome
SAVE «HELLO»	Memorizza sul nastro col nome HELLO
SAVE A$	Memorizza sul nastro col nome della variabile A$

SAVE «HELLO»,8	Memorizza sul disco col nome HELLO
SAVE «HELLO»,1,1	Memorizza sul nastro col nome HELLO e fa seguire il programma da un marcatore di fine nastro (END-OF-TAPE).

VERIFY

Questo comando fa sì che il VIC controlli il programma su nastro o su disco a fronte di quello esistente in memoria. Questa è la prova che il programma che è stato recentemente salvato (SAVE) lo è realmente nel caso in cui il nastro sia difettoso o qualcosa non funzioni. Questo comando è anche molto utile per posizionare un nastro in modo che il VIC scriva *dopo* l'ultimo programma presente sul nastro stesso. Tutto ciò che occorre fare è di dire al VIC di VERIFY (verificare) il nome dell'ultimo programma sul nastro. Esso lo farà e dirà che i programmi non corrispondono (cosa che già si sapeva). Ora il nastro è nel punto in cui si vuole ed è possibile memorizzare il successivo programma senza timore di cancellarne uno vecchio.

VERIFY seguito da nulla fa sì che il VIC controlli il successivo programma sul nastro, indipendentemente dal suo nome, a fronte del programma che si trova ora in memoria. VERIFY seguito da un nome di programma (tra virgolette) o da una variabile stringa cercherà nel nastro quel programma e quindi lo controllerà. VERIFY seguito da un nome e da una virgola e da un numero controllerà il programma sul dispositivo con quel numero (1 per il nastro, 8 per il disco).

ESEMPIO:

VERIFY	Controlla il successivo programma sul nastro
VERIFY «HELLO»	Cerca HELLO, controlla a fronte della memoria
VERIFY «HELLO»,8	Cerca HELLO sul disco, quindi controlla

3. Istruzioni

CLOSE

Questo comando completa e chiude qualsiasi file usato dalle istruzioni OPEN. Il numero che segue la parola CLOSE è il numero del file da chiudere.

ESEMPIO:

CLOSE 2 Viene chiuso soltanto il file N. 2

CLR

Questo comando cancella qualsiasi variabile di memoria ma lascia il programma intatto. Viene automaticamente eseguito quando viene impartito un comando RUN.

CMD

CMD invia l'output che normalmente andrebbe allo schermo (e cioè le istruzioni PRINT, LIST ma non POKE) su un altro dispositivo. Questo dispositivo potrebbe essere una stampante, un file di dati su nastro o su disco. Questo dispositivo o file deve essere per prima cosa aperto (OPEN). Il comando CMD deve essere seguito da un numero o da una variabile numerica che fa riferimento al file.

ESEMPIO:

OPEN 1,4	Apre il dispositivo N. 4 che è la stampante
CMD 1	Tutto l'output normale va ora alla stampante
LIST	Il comando indirizza alla stampante e non allo schermo – anche la parola LIST che è stata battuta!

Per riprendere il normale invio allo schermo, basta chiudere (CLOSE) il file.

DATA

Questa istruzione è seguita da un elenco di elementi da usarsi dalle istruzioni READ. Questi elementi possono essere numeri o parole e sono separate da virgole. Le parole non devono essere all'interno di virgolette a meno che esse non contengano uno dei seguenti caratteri: SPACE (spazio), virgola (,) o due punti (:). Se ci sono due virgole senza nulla all'interno, il valore verrà letto (READ) come zero se si tratta di un numero o come stringa vuota.

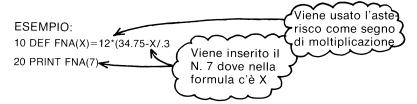
ESEMPIO DI UN'ISTRUZIONE DATA:

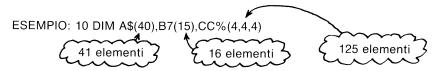
DATA 100,200,FRED,«HELLO, MOM»,3.14,abc123

Dato che il programma non deve mai eseguire effettivamente un'istruzione DATA per leggere le informazioni, è una buona idea inserire le istruzioni DATA il più vicino possibile all'ultima riga del programma. Ciò fa sì che i programmi girino più velocemente.

DEF FN (Definisci Funzione)

Questo comando consente di definire un calcolo complesso come una funzione con un breve nome. Nel caso di una lunga formula che è usata parecchie volte durante il programma ciò può far risparmiare molto spazio.

Il nome che si attribuisce ad una funzione sarà costituito dalle lettere FN e da qualsiasi nome variabile lecito (lungo 1 o 2 caratteri). Innanzitutto occorre definire la funzione usando l'istruzione DEF seguita dal nome attribuito alla funzione stessa. Dopo il nome c'è una serie di parentesi (=) con una variabile numerica (in questo caso X) al loro interno. Quindi c'è un segno di uguale seguito dalla formula che si vuole definire. E' possibile «richiamare» la formula sostituendo qualsiasi numero a X, usando il formato indicato nella riga 20 dell'esempio che segue:

DIM (Dimensionamento di una matrice)

Prima di poter usare matrici di variabili, a meno che non abbiamo 11 elementi o meno, il programma deve per prima cosa eseguire un'istruzione DIM per dimensionare quella matrice. L'istruzione DIM è seguita dal nome della matrice, che può essere qualsiasi nome variabile lecito. Quindi, racchiuso tra parentesi, occorre inserire il numero (una variabile numerica) di elementi in ciascuna dimensione. E' possibile usare qualsiasi numero di dimensioni ma occorre tenere presente che l'intera lista di variabili che si crea occupa molto spazio ed è facile esaurire in questo caso la disponibilità di memoria. Per immaginare il numero di variabili create con ciascun DIM, basta moltiplicare il numero totale di elementi in ciascuna dimensione della matrice.

E' possibile dimensionare più di una matrice in un'istruzione DIM separando le varie matrici mediante virgole. Fare attenzione a non permettere al programma di eseguire un'istruzione DIM per una matrice più di una volta altrimenti si ha un messaggio di errore. E' una buona idea tenere le istruzioni DIM vicine all'inizio del programma.

END

Quando il programma incontra una riga con l'istruzione END, interrompe l'esecuzione come se avesse esaurito tutte le righe. E' possibile usare il comando CONT per far ripartire il programma.

FOR . . . TO . . . STOP

Questa istruzione funziona con l'istruzione NEXT per creare una sezione del programma che si ripete per un certo numero di volte. E' possibile fare in modo che il VIC conti fino ad un grande numero in modo che il programma faccia una pausa per pochi secondi oppure è possibile contare qualcosa. Queste sono le istruzioni più comunemente usate nel BASIC.

Il formato dell'istruzione è il seguente:

FOR (nome variabile iterazione) = (inizio del conteggio) TO (fine del conteggio). La variabile di iterazione è una variabile che verrà aggiunta o sottratta durante il programma. L'inizio del conteggio e la fine del conteggio sono i limiti al valore della variabile di iterazione.

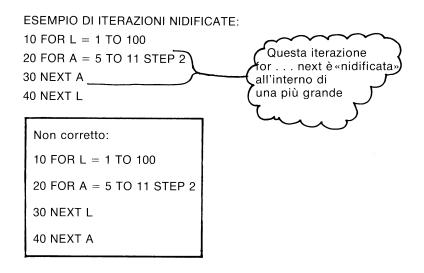
La logica dell'istruzione FOR è la seguente. Innanzitutto viene definita la variabile di iterazione all'inizio del valore di conteggio. La fine del valore di conteggio viene salvata per successivo riferimento da parte del VIC. Quando il programma raggiunge una riga con il comando NEXT, aggiunge uno al valore della variabile di iterazione e controlla se questo è più alto del valore di fine iterazione. Se non è più alto, la successiva riga eseguita è l'istruzione immediatamente seguente l'istruzione FOR. Se la variabile di iterazione è maggiore del numero di fine dell'iterazione, la successiva istruzione eseguita sarà quella che segue l'istruzione NEXT.

ESEMPIO:

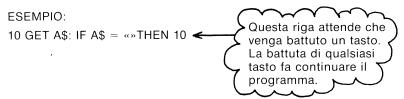
```
10 FOR L = 1 TO 10
20 PRINT L
30 NEXT L
40 PRINT «I'M DONE! L =»L
```

Questo programma stamperà sullo schermo i numeri da uno a dieci, seguiti dal messaggio I'M DONE! L=11 (Fatto! L=11). Visto come funziona? In caso contrario cercare di rileggere il paragrafo prima dell'esempio e ricostruire il programma una fase alla volta sulla carta.

Il valore di fine iterazione può essere seguito dalla parola STEP e da un altro numero o variabile. In questo caso viene aggiunto ogni volta il valore che segue STEP invece di uno. Ciò consente di contare all'indietro ossia a decrementare, per frazioni o in qualsiasi modo necessario.

E' possibile creare iterazioni una all'interno dell'altra. Il procedimento è noto come *nidificazione.* Occorre però fare attenzione a nidificare le iterazioni in modo che l'iterazione che inizia per ultima sia quella che termina per prima.

GET

L'istruzione GET è un modo per ottenere i dati dalla tastiera un carattere alla volta. Quando viene eseguito GET, viene ricevuto il carattere che era stato battuto. Se non viene battuto un carattere si riceve un carattere nullo (vuoto) ed il programma continua. Non c'è necessità di battere il tasto RETURN ed in effetti il tasto RETURN può essere ricevuto con un GET.

Il termine GET è seguito da un nome variabile, solitamente una variabile stringa. Se fosse usato un valore numerico e fosse battutto qualsiasi altro tasto diverso dal numero, il programma si interromperebbe con un messaggio di errore. L'istruzione GET può anche essere inserita in un'iterazione, alla ricerca di un risultato vuoto, in attesa della battuta di un tasto.

GET#

Usato con un dispositivo o file precedentemente aperto (OPEN) per immettere un carattere alla volta.

ESEMPIO: GET#1, A$

GOSUB

Questa istruzione è come l'istruzione GOTO salvo che il VIC ricorda da dove proviene. Quando si incontra una riga con un'istruzione RETURN, il programma salta indietro all'istruzione che segue direttamente GOSUB. Ciò è utile quando c'è una routine nel programma che si verifica parecchie volte in diverse parti del programma stesso. Invece di batterla ripetutamente, la si batte una volta e si ritorna su di essa (GOSUB) dalle diverse parti del programma. 20 GOSUB 800 significa «andare alla subroutine che inizia alla riga 800 ed eseguirla».

GOTO o GO TO

Quando viene raggiunta un'istruzione con il comando GOTO, la riga successiva da eseguire sarà quella con il numero che segue la parola GOTO.

IF . . . THEN

L'istruzione IF . . . THEN consente al VIC di analizzare una situazione ed intraprendere due azioni possibili in relazione al risultato. Se l'espressione viene valutata e risulta vera, viene eseguita l'istruzione che segue la parola THEN. Questa può essere un numero di riga, che fa sì che il VIC vada (GOTO) a quella riga del programma. Può anche essere qualsiasi altra istruzione o istruzioni BASIC. Se l'espressione è falsa, viene invece eseguita la riga successiva (e non la successiva istruzione sulla stessa riga).

L'espressione che viene valutata può essere una variabile o una formula, nel qual caso è considerata vera se diversa da zero e falsa se uguale a zero. Nella maggior parte dei casi, c'è un'espressione che coinvolge gli *operatori relazionali* (=,<,>,AND,OR,NOT). Se il risultato risulta vero, ha un valore di –1 ed un valore di 0 se falso. Vedere la parte sugli operatori relazionali per una spiegazione del modo in cui funzionano.

INPUT

L'istruzione INPUT consente al computer di trasferire dati in una variabile. Il programma si interrompe, stampa un punto di domanda (?) sullo schermo ed attende che l'operatore batta la risposta e prema il tasto RETURN.

La parola INPUT è seguita da un nome variabile o lista di nomi variabili separati da virgole. Prima della lista di variabili da immettere può esserci un messaggio all'interno delle virgolette. Se questo messaggio (detto *richiesta*) è presente, deve esserci un punto e virgola (;) dopo le ultime virgolette della richiesta. Quando deve essere immessa (INPUT) più di una variabile, le variabili devono essere separate da virgole quando vengono battute.

ESEMPIO:

```
10 INPUT«PLEASE TYPE A#»;A
20 INPUT«AND YOUR NAME»;A$
30 INPUT B$
40 PRINT«BET YOUR DIDN'T KNOW  WHAT I
WANTED!»
```

INPUT#

Questa funziona come INPUT, ma prende i dati da un file o dispositivo precedentemente aperti (OPEN).

LET

La parola LET difficilmente è usata nei programmi dato che è facoltativa ma l'istruzione che la contiene è il cuore dei programmi BASIC. Il nome variabile che deve acquisire il risultato di un calcolo si trova sul lato sinistro del segno di uguale ed il numero o formula sul lato destro.

ESEMPIO:

```
10 LET A=5
20 B=6
30 C=A*B+3
40 D$=«HELLO»
```

NEXT

L'istruzione NEXT è sempre usata unitamente all'istruzione FOR. Quando il programma arriva ad un'istruzione NEXT, ritorna indietro all'istruzione FOR e controlla l'iterazione. (Vedere per ulteriori dettagli l'istruzione FOR). Se l'iterazione è terminata, l'esecuzione procede con l'istruzione che segue NEXT. La parola NEXT può essere seguita da un nome variabile o da una lista di nomi variabili, separati da virgole. Se non sono elencati nomi, l'ultima iterazione iniziata à quella che viene completata. Se vengono indicate variabili, sono completate nell'ordine da sinistra a destra.

ESEMPIO:
```
10 FOR L=1 TO 10:NEXT
20 FOR L=1 TO 10:NEXT L
30 FOR L=1 TO 10:FOR M=1 TO 10:NEXT M,L
```

ON

Questo comando può introdurre i comandi GOTO e GOSUB in versioni speciali dell'istruzione IF. La parola ON è seguita da una formula, che dà per risultato un numero. La parola GOTO o GOSUB è seguita da una lista di numeri separati da virgole. Se il risultato del calcolo è 1, viene eseguita la prima riga nella lista. Se il risultato è 2, viene eseguita la seconda riga e così via. Se il risultato è 0, negativo o maggiore dei numeri di riga, la successiva istruzione eseguita sarà quella che segue le istruzioni ON.

ESEMPIO:
```
10 INPUT X
20 ON X GOTO 10,50,50,50
30 PRINT«NOPE!»
40 GOTO 10
50 PRINT«YUP!»
60 ON X GOTO 10,30,30
```

OPEN

L'istruzione OPEN consente al VIC di accedere ai dispositivi, ad esempio il registratore a cassetta e l'unità disco per i dati, una stampante o addirittura lo schermo del VIC. La parola OPEN è seguita da un numero che è quello al quale faranno riferimento tutte le altre istruzioni BASIC. Questo numero va da 1 a 255. Dopo il primo c'è subito il secondo numero, separato da una virgola. Questo è il numero del dispositivo, 0 per lo schermo del VIC, 1 per il registratore a cassetta, 4 per la stampante, 8 per il disco. E' buona idea usare lo stesso numero di riferimento come numero del dispositivo il che facilita l'identificazione. Dopo il secondo numero può essere aggiunto un terzo numero, sempre separato da una virgola, che è l'indirizzo secondario. Nel caso della cassetta, questo può essere 0 per la lettura, 1 per la scrittura e 2 per la scrittura con marcatore di fine nastro al termine. Nel caso del disco, il numero si riferisce al numero del buffer o del canale. Nella stampante l'indirizzo secondario si trasforma in diversi tipi di comandi. Vedere il Manuale di Riferimento del Programmatore VIC per maggiori

chiarimenti. Può esserci anche una stringa che segue il terzo numero, che potrebbe rappresentare un comando all'unità disco o il nome del file sul nastro.

ESEMPIO:

10 OPEN 1,0	Apre (OPEN) lo schermo come un dispositivo
20 OPEN 2,1,0,«D»	Apre (OPEN) la cassetta per la lettura; il file da cercare è denominato D
30 OPEN 3,4	Per usare la stampante
40 OPEN 4,8,15	Apre (OPEN) il canale dei dati sul disco

Vedere anche le istruzioni CLOSE, CMD, GET#, INPUT# e PRINT#, variabili di sistema ST ed Appendice B.

POKE

Il comando POKE è sempre seguito da due numeri o formule. Il primo numero è una locazione all'interno della memoria del VIC. Potrebbero esserci locazioni numerate da 0 a oltre 65.000. Alcune di queste, come quelle descritte nei capitoli sul suono e sui colori, possono essere usate facilmente nei programmi. Altre, per contro, sono usate dal VIC per creare dei programmi e così via. La sperimentazione con l'istruzione POKE probabilmente darà luogo ad effetti interessanti. Se succede qualcosa e non è possibile interrompere, basta spegnere il VIC e riaccenderlo o tenere abbassato il tasto RUN/STOP e premere RESTORE.

Il secondo numero è un valore da 0 a 255 che verrà inserito nella locazione di memoria, sostituendo qualsiasi valore inserito precedentemente.

ESEMPIO:

```
10 POKE 36879,8
20 POKE 9*16↑ 3+15,27
```

PRINT

L'istruzione PRINT è la prima che la tutti imparano a conoscere e ad usare ma occorre conoscerne le numerose sottigliezze. La parola PRINT può essere seguita da:

Parole tra le virgolette

Nomi variabili

Funzioni

Segni di punteggiatura

Le parole all'interno delle virgolette sono spesso dette *costanti* in quanto sono stampate letteralmente come vengono battute. I nomi variabili al di fuori delle virgolette compariranno col valore che contengono. Anche le funzioni avranno i rispettivi valori stampati. I segni di punteggiatura sono usati per aiutare a formattare ordinatamente i dati sullo schermo. La virgola è usata per dividere lo schermo in due colonne, mentre il punto e virgola non lascia alcun spazio. Uno o l'altro segno di punteggiatura può essere usato come ultimo simbolo nell'istruzione. Ciò fa sì che la successiva cosa

stampata (PRINT) risulti come se fosse la continuazione della stessa istruzione PRINT.

ESEMPIO:
```
10 PRINT «HELLO»
20 PRINT «HELLO,»A$
30 PRINT A+B;
50 PRINT J;
60 PRINT A,B,C,D
```

Vedere anche le funzioni POS(), SPC(), TAB().

PRINT#

Ci sono alcune differenze tra questa istruzione e l'istruzione PRINT. Innanzitutto la parola PRINT# è seguita da un numero che si riferisce al dispositivo o file di dati precedentemente aperto (OPEN). Il numero è seguito da una virgola e da una lista di cose da stampare (PRINT). La virgola ed il punto e virgola hanno lo stesso effetto dell'aggiunta di spazi come del resto fanno nell'istruzione PRINT, ma alcuni dispositivi possono non funzionare con TAB e SPC.

ESEMPIO:
```
100 PRINT#1, «HELLO THERE!»;A$,B$
```

READ

Questa istruzione viene usata per inserire nelle variabili informazioni dalle istruzioni DATA, dove possono essere usate. Si deve fare attenzione a non leggere stringhe dove l'istruzione READ chiede un numero, per evitare la comparsa del messaggio TYPE MISMATCH ERROR.

REM (Nota)

REM non è altro che un commento a chiunque stia leggendo una lista (LIST) del programma. Può spiegare una sezione del programma, fornire informazioni sull'autore, ecc. Le istruzioni REM non interessano in alcun modo il funzionamento del programma, solo lo prolungano. La parola ROM può essere seguita da qualsiasi testo, quantunque l'uso di caratteri grafici dia risultati strani (vedere per ulteriori dettagli la Guida di Riferimento del Programmatore VIC).

RESTORE

Eseguita in un programma, il puntatore al quale l'elemento nell'istruzione DATA verrà letto successivamente è ripristinato al primo elemento della lista. Ciò consente di rileggere le informazioni. La parola RESTORE sta da sola sulla riga.

RETURN

Questa istruzione è sempre usata unitamente all'istruzione GOSUB. Quando il programma incontra un'istruzione RETURN, si porta all'istruzione che segue immediatamente GOSUB. Se non era stato emesso precedentemente alcun GOSUB, c'è un messaggio RETURN WITHOUT GOSUB ERROR (Errore RETURN senza GOSUB). Nulla segue la parola RETURN.

STOP

Questa istruzione interrompe il programma. Compare un messaggio, BREAK ERROR IN LINE xxxx (Errore di interruzione alla riga xxxx), dove xxxx è il numero di riga contenente il comando STOP. Il programma può essere fatto ripartire usando il comando CONT. L'istruzione STOP è usata per la correzione degli errori di un programma.

SYS

La parola SYS è seguita da un numero decimale o da una variabile numerica nel campo da 0 a 65535. Il programma a questo punto inizia l'esecuzione del programma in linguaggio macchina iniziando da quella locazione di memoria. E' simile alla funzione USR, ma non consente di passare parametri.

WAIT

L'istruzione WAIT è usata per interrompere il programma fino a che i contenuti di una locazione di memoria non cambiano in un modo specifico. La parola WAIT è seguito da un numero, che è l'indirizzo di memoria che viene controllato. Vengono quindi una virgola ed un altro numero. Possono esserci anche un'altra virgola ed un terzo numero. Questi ultimi due numeri devono essere nel campo da 0 a 255.

I contenuti della locazione di memoria sono per prima cosa sottoposti ad un'operazione di OR-esclusivo con il terzo numero, se presente e quindi logicamente sottoposti ad un'intersezione logica (AND) con il secondo numero. Se il risultato è zero, il programma ritorna a quella locazione di memoria e controlla di nuovo. Quando il risultato è diverso da zero, il programma continua con la successiva istruzione.

4. Funzioni

a. Numeriche

ABS(X) (valore assoluto)

Il valore assoluto dà il valore del numero, senza il segno (– o +). Il risultato è sempre positivo.

ATN(X) (arcotangente)

Dà l'angolo, misurato in radianti, la cui tangente è X.

COS(X) (coseno)

Dà il valore del coseno di X, dove X è l'angolo misurato in radianti.

EXP(X)

Dà il valore della costante matematica *e* (2.71827183) elevata alla potenza di X.

FNXX(X)

Dà il valore della funzione XX definita dall'utente che apre in un'istruzione DEF FNXX.

INT(X) (intero)

Dà il valore troncato di X e cioè con tutte le cifre decimali alla destra del punto decimale rimosse. Il risultato sarà sempre minore di o uguale a X. Pertanto qualsiasi numero negativo con cifre decimali diventa l'intero *minore* del suo valore corrente.
Se la funzione INT deve essere usata per l'arrotondamento per eccesso o per difetto, la forma è INT (X + .5).

ESEMPIO:

X = INT(X*100 + .5)/100 Arrotonda al centesimo più vicino

LOG(X) (logaritmo)

Questa funzione dà il logaritmo naturale di X. Il logaritmo naturale è il logaritmo in base *e* (vedere EXP(X). Per convertire nel logaritmo in base 10, basta semplicemente dividere per LOG(10).

PEEK(X)

Questa funzione è usata per trovare i contenuti della locazione di memoria X nel campo da 0 a 65535 ottenendo un risultato compreso tra 0 e 255. E' spesso usata unitamente all'istruzione POKE.

RND(X) (numero casuale)

Questa funzione dà un numero casuale (o pseudo casuale) compreso tra 0 e 1. E' utile nei giochi per simulare il lancio di dadi ed altri elementi di probabilità ed è anche usata in alcune applicazioni statistiche. Il primo numero casuale deve essere generato dalla formula RND(–TI), per partire ogni volta con un numero diverso. Dopo di ciò, il numero in X dovrebbe essere un 1 o qualsiasi numero positivo. Se X è zero, il risultato sarà lo stesso numero casuale precedentemente uscito. Un valore negativo di X risemina il generatore. L'uso dello stesso numero negativo per X dà luogo alla stessa sequenza di numeri «casuali».

Per simulare il lancio di un dado, usare la formula INT(RND(1)*6+1). Per prima cosa il numero casuale da 0 a 1 è moltiplicato per 6 che espande il campo da 0-6 (in effetti maggiore di zero e minore di sei). Quindi viene aggiunto 1, in modo che il campo risulti di 1 inferiore a 7. La funzione INT toglie tutte le cifre decimali lasciando come risultato una cifra da 1 a 6.

Per simulare 2 dadi, sommare due dei numeri ottenuti dalla suddetta formula.

ESEMPIO:

100 X = INT(RND(1)*6)+INT(RND(1)*6)+2	Simula il lancio di due dadi
100 X = INT(RND(1)*1000)+1	Numero da 1 a 1000
100 X + INT(RND(1)*150)+100	Numero da 100 a 249

SGN(X) (segno)

Questa funzione dà il segno, positivo, negativo o zero di X. Il risultato sarà +1 se positivo, 0 se zero e –1 se negativo.

SIN(X) (seno)

Questa è la funzione trigonometrica seno. Il risultato sarà il seno di X, dove X è un angolo in radianti.

SQR(X) (radice quadrata)

Questa funzione darà la radice quadrata di X dove X è un numero positivo o 0. Se X è negativo, si ottiene un messaggio ILLEGAL QUANTITY ERROR (Errore quantità illecita).

TAN(X) (tangente)

Il risultato sarà la tangente di X, dove X è un angolo in radianti.

USR(X)

Quando viene usata questa funzione, il programma salta ad un programma in linguaggio macchina il cui punto di partenza è contenuto nelle locazioni di memoria 1 e 2. Il parametro X viene passato al programma in linguaggio macchina che ritorna un altro numero al programma BASIC. Vedere per ulteriori dettagli su questo argomento e sulla programmazione in linguaggio macchina il Manuale di Riferimento del Programmatore VIC.

b. Funzioni stringa

ASC(X$)

Questa funzione dà il codice ASCII del primo carattere di X$.

CHR$(X)

Questa è l'opposto di ASC e dà un carattere stringa il cui codice ASCII è X.

LEFT$(X$,X)

Ciò dà una stringa contenente i caratteri X più a sinistra di X$.

LEN(X$)

Dà il numero di caratteri (compresi gli spazi ed altri simboli) nella stringa X$.

MID$(X$,S,X)

Questa funzione darà una stringa contenente X caratteri iniziando dall'S-esimo carattere in X$.

RIGHT$(X$,X)

Questa funzione darà i caratteri X più a destra in X$.

STR$(X)

Questa funzione conterrà una stringa che è identica alla versione stampata (PRINT) di X$.

VAL(X$)

Questa funzione converte la stringa X$ in un numero ed è essenzialmente l'operazione inversa di STR$. La stringa viene esaminata dal carattere più a sinistra fino a quello più a destra alla ricerca di quanti sono i caratteri in formato numerico riconoscibili. Se il VIC trova caratteri illeciti, viene convertita soltanto la porzione della stringa fino a quel punto.

ESEMPIO:	10 X = VAL(«123.456)	X = 123.456
	10 X = VAL(«12A13B»)	X = 12
	10 X = VAL(«RIUO17*»)	X = 0
	10 X = VAL(«–1.23.23.23»)	X = –1.23

c. Altre Funzioni

FRE(X)

Questa funzione dà il numero di byte inutilizzati disponibili in memoria, indipendentemente dal valore di X.

POS(X)

Questa funzione dà il numero della colonna (0-21) al quale inizierà la successiva istruzione PRINT sullo schermo. X può avere qualsiasi valore e non è usato.

SPC(X)

Questa funzione è usata nell'istruzione PRINT per saltare X spazi in avanti.

TAB(X)

Questa funzione è usata nell'istruzione PRINT. La successiva voce da stampare sarà nella colonna numero X.

Appendice D:
Abbreviazioni per le parole chiave BASIC

Come metodo per risparmiare tempo nel battere programmi e comandi, il BASIC VIC consente all'utente di abbreviare la maggior parte delle parole chiave. L'abbreviazione per la parola PRINT è un punto di domanda. L'abbreviazione per le altre parole sono ottenute battendo la seconda delle prime due lettere della parola chiave tenendo premuto contemporaneamente il tasto SHIFT. Se in una riga di programma sono usate abbreviazioni, la parola chiave le listerà (LIST) nella forma più lunga. Notare che alcune delle parole chiave quando abbreviate comprendono la prima parentesi, altre no.

Comando	Abbrev.	Come appare su schermo
AND	A SHIFT N	A ╱
NOT	N SHIFT O	N □
CLOSE	CL SHIFT O	CL □
CLR	C SHIFT L	C □
CMD	C SHIFT M	C ╲
CONT	C SHIFT O	C □
DATA	D SHIFT A	D ♠
DEF	D SHIFT E	D ▔
DIM	D SHIFT I	D ╮
END	E SHIFT N	E ╱
FOR	F SHIFT O	F □
GET	G SHIFT E	G ▔
GOSUB	GO SHIFT S	GO ♥
GOTO	G SHIFT O	G □
INPUT#	I SHIFT N	I ╱
LET	L SHIFT E	L ▔
LIST	L SHIFT I	L ╮
LOAD	L SHIFT O	L □
NEXT	N SHIFT E	N ▔
OPEN	O SHIFT P	O □
POKE	P SHIFT O	P □
PRINT	?	?
RND	R SHIFT N	R ╱
SGN	S SHIFT G	S ▏
SIN	S SHIFT I	S ╮
SPC (	S SHIFT P	S □
SQR	S SHIFT Q	S ●
PRINT#	P SHIFT R	P ▁
READ	R SHIFT E	R ▔
RESTORE	RE SHIFT S	RE ♥
RETURN	RE SHIFT T	RE ▏
RUN	R SHIFT U	R ╭
SAVE	S SHIFT A	S ♠
STEP	ST SHIFT E	ST ▔
STOP	S SHIFT T	S ▏
SYS	S SHIFT Y	S ▕
THEN	T SHIFT H	T ▕
VERIFY	V SHIFT E	V □
WAIT	W SHIFT A	W ♠
ABS	A SHIFT B	A │
ASC	A SHIFT S	A ♥
ATN	A SHIFT T	A ▏
CHR$	C SHIFT H	C ▕
EXP	E SHIFT X	E ♣
FRE	F SHIFT R	F ▁
LEFT$	LE SHIFT F	LE ▁
MID$	M SHIFT I	M ╮
PEEK	P SHIFT E	P ▔
RIGHT$	R SHIFT I	R ╮
STR$	ST SHIFT R	ST ▁
TAB (	T SHIFT A	T ♠
USR	U SHIFT S	U ♥
VAL	V SHIFT A	V ♠

Appendice E:
Combinazioni di colori dello schermo e del margine

E' possibile cambiare i colori dello schermo e del margine del VIC in qualsiasi momento, all'interno o all'esterno di un programma, battendo

POKE 36879,X

dove X è uno dei numeri indicati sulla tabella che segue. POKE 36879,27 riporta lo schermo alla normale combinazione dei colori, che è un margine blu verde e lo schermo bianco.

Provare a battere POKE 36879,8. Quindi battere CTRL 2 WHT per avere lettere bianche su uno schermo totalmente nero! Provare qualche altra combinazione. Questo comande POKE è un modo semplice e rapido per cambiare i colori dello schermo in un programma.

Margine

	BLK	WHT	RED	CYAN	PUR	GRN	BLU	YEL
SCHERMO	**NERO**	**BIANCO**	**ROSSO**	**BLU-VERDE**	**PORPORA**	**VERDE**	**BLU**	**GIALLO**
NERO	8	9	10	11	12	13	14	15
BIANCO	24	25	26	27	28	29	30	31
ROSSO	40	41	42	43	44	45	46	47
BLU-VERDE	56	57	58	59	60	61	62	63
PORPORA	72	73	74	75	76	77	78	79
VERDE	88	89	90	91	92	93	94	95
BLU	104	105	106	107	108	109	110	111
GIALLO	120	121	122	123	124	125	126	127
ARANCIO	136	137	138	139	140	141	142	143
ARANCIOCHI.	152	153	154	155	156	157	158	159
ROSA	168	169	170	171	172	173	174	175
BLU-VERDE CHI.	184	185	186	187	188	189	190	191
PORPORACHI.	200	201	202	203	204	205	206	207
VERDECHIARO	216	217	218	219	220	221	222	223
AZZURRO	232	233	234	235	236	237	238	239
GIALLOCHI.	248	249	250	251	252	253	254	255

Appendice F: Tabella delle note musicali

Nota approssimat.	Valore	Nota approssimat.	Valore
C	135	G	215
C#	143	A^b	217
D	147	A	219
E^b	151	B	221
E	159	H	223
F	163	C	225
F#	167	C#	227
G	175	D	228
A^b	179	E^b	229
A	183	E	231
B	187	F	232
H	191	F#	233
C	195	G	235
C#	199	A^b	236
D	201	A	237
E^b	203	B	238
E	207	H	239
F	209	C	240
F#	212	C#	241

Nota: Corrispondenza delle note in notazione anglosassone e italiana: C = Do, D = Re, E = Mi, F = Fa, G = Sol, A = La, B = Si

Comandi altoparlante	Dove X può essere	Funzione
POKE 36878,X	Da 0 a 15	Regola volume
POKE 36874,X	Da 128 a 255	Suona un tono
POKE 36875,X	Da 128 a 255	Suona un tono
POKE 36876,X	Da 128 a 255	Suona un tono
POKE 36877,X	Da 128 a 255	Suona «rumore»

Appendice G: 20 effetti sonori per il VIC-20

Ecco alcune semplici routines da usare come guida per creare suoni per ravvivare i programmi. E' possibile batterle nel VIC-20 sia isolatamente che all'interno di altri programmi. Naturalmente questi non sono tutti i possibili suoni che il VIC-20 può eseguire percui c'è la possibilità di mettere alla prova la propria creatività.

Gli effetti sonori qui elencati fanno sì che il programma faccia una pausa necessaria per permetterne il completamento. E' possibile inserire questi effetti in un programma in modo che non interrompa qualsivoglia animazione possa essere in corso ma questo argomento è discusso in dettaglio nel Manuale di Riferimento del Programmatore del VIC-20.

Ricordarsi di usare il numero di riga quando si battono queste routines nel computer. I numeri non sono indicati qui in ordine per evitare confusione quando li si immette nei programmi.

#1: SCALE

```
POKE 36878,15
FOR L=250 TO 200 STEP -2
POKE 36876,L
FOR M=1 TO 100
NEXT M
NEXT L
FOR L=205 TO 250 STEP 2
POKE 36876,L
FOR M=1 TO 100
NEXT M
NEXT L
POKE 36876,0
POKE 36878,0
```

#2: COMPUTERMANIA

```
POKE 36878,15
FOR L=1 TO 100
POKE 36876,INT(RND(1)*128)+128
FOR M=1 TO 10
NEXT M
NEXT L
POKE 36876,0
POKE 36878,0
```

#3: ESPLOSIONE

```
POKE 36877,220
FOR L=15 TO 0 STEP -1
POKE 36878,L
FOR M=1 TO 300
NEXT M
NEXT L
POKE 36877,0
POKE 36878,0
```

#4: CADUTA DI BOMBE

```
POKE 36878,10
FOR L=230 TO 128 STEP -1
POKE 36876,L
FOR M=1 TO 20
NEXT M
NEXT L
POKE 36876,0
POKE 36877,200
FOR L=15 TO 0 STEP -.05
POKE 36878,L
NEXT L
POKE 36877,0
```

#5: ALLARME ROSSO

```
POKE 36878,15
FOR L=1 TO 10
FOR M=180 TO 235 STEP 2
POKE 36876,M
FOR N=1 TO 10
NEXT N
NEXT M
POKE 36876,0
FOR M=1 TO 100
NEXT M
NEXT L
POKE 36878,0
```

#6: RAGGIO LASER

```
POKE 36878,15
FOR L=1 TO 30
FOR M=250 TO 240 STEP -1
POKE 36876,M
NEXT M
FOR M=240 TO 250
POKE 36876,M
NEXT M
POKE 36876,0
NEXT L
POKE 36878,0
```

#7: SIRENA BITONALE

```
POKE 36878,15
FOR L=1 TO 10
POKE 36875,200
FOR M=1 TO 500
NEXT M
POKE 36875,0
POKE 36876,200
FOR M=1 TO 500
NEXT M
POKE 36876,0
NEXT L
POKE 36878,0
```

#8: SEGNALE DI OCCUPATO

```
POKE 36878,15
FOR L=1 TO 15
POKE 36876,160
FOR M=1 TO 400
NEXT M
POKE 36876,0
FOR M=1 TO 400
NEXT M
NEXT L
POKE 36878,0
```

#9: SUONO DEL TELEFONO

```
POKE 36878,15
FOR L=1 TO 5
FOR M=1 TO 50
POKE 36876,220
FOR N=1 TO 5
NEXT N
POKE 36876,0
NEXT M
FOR M=1 TO 3000
NEXT M
NEXT L
POKE 36878,0
```

#10: PIGOLIO DI UCCELLI

```
POKE 36878,15
FOR L=1 TO 20
FOR M=254 TO 240+INT (RND(1)*10) STEP -1
POKE 36876,M
NEXT M
POKE 36876,0
FOR M=0 TO INT(RND(1)*100)+120
NEXT M
NEXT L
```

#11: VENTO

```
POKE 36878,15
POKE 36874,170
POKE 36877,240
FOR L=1 TO 2000
NEXT L
POKE 36874,0
POKE 36877,0
POKE 36878,0
```

#12: ONDE DELL'OCEANO

```
POKE 36877,180
FOR L=1 TO 10
D=INT(RND(1)*5)*50+50
FOR M=3 TO 15
POKE 36878,M
FOR N=1 TO D
NEXT N
NEXT M
FOR M=15 TO 3 STEP -1
POKE 36878,M
FOR N=1 TO D
NEXT N
NEXT M
NEXT L
POKE 36878,0
POKE 36877,0
```

#13: UFO CHE SCOMPARE

```
POKE 36878,15
FOR L=130 TO 254
POKE 36876,L
FOR M=1 TO 40
NEXT M
NEXT L
POKE 36878,0
POKE 36876,0
```

#14: ATTERRAGGIO DI UN UFO

```
POKE 36878,15
FOR L=1 TO 20
FOR M=220-L TO 160-L STEP -4
POKE 36876,M
NEXT M
FOR M=160-L TO 220-L STEP 4
POKE 36876,M
NEXT M
NEXT L
POKE 36878,0
POKE 36876,0
```

#15: UFO CHE SPARA

```
POKE 36878,15
FOR L=1 TO 15
FOR M=200 TO 220+L*2
POKE 36876,M
NEXT M
NEXT L
POKE 36878,0
POKE 36876,0
```

#16: ULULO DEL LUPO

```
POKE 36878,15
FOR L=148 TO 220 STEP .7
POKE 36876,L
NEXT L
FOR L=128 TO 200
POKE 36876,L
NEXT L
FOR L=200 TO 128 STEP -1
POKE 36876,L
NEXT L
POKE 36878,0
POKE 36876,0
```

#17: RUMORE DI PIEDI

```
POKE 36878,15
FOR L=1 TO 10
POKE 36874,200
FOR M=1 TO 10
NEXT M
POKE 36874,0
FOR M=1 TO 100
NEXT M
NEXT L
POKE 36878,0
```

#18: TICK-TOCK

```
POKE 36878,15
FOR L=1 TO 10
POKE 36875,200
FOR M=1 TO 10
NEXT M
POKE 36875,0
FOR M=1 TO 300
NEXT M
POKE 36874,200
FOR M=1 TO 10
NEXT M
POKE 36874,0
FOR M=1 TO 300
NEXT M
NEXT L
POKE 36878,0
```

#19: APERTURA DI UNA PORTA

```
POKE 36878,15
B=0
FORL=128 TO 255 STEP 11
POKE 36874,L
FOR M=1 TO 10
NEXT M
B=B+1
IF B=3 THEN B=0: POKE 36874,0
NEXT L
POKE 36874,0
POKE 36878,0
```

#20: BLIP

```
POKE 36878,15
POKE 36876,220
FOR L=1 TO 5
NEXT L
POKE 36876,0
FOR L=1 TO 500
NEXT L
POKE 36876,200
FOR L=1 TO 5
NEXT L
POKE 36876,0
FOR L=1 TO 500
NEXT L
POKE 36878,0
```

Appendice H: Codici dello schermo

La tabella che segue elenca tutti i caratteri incorporati nella serie di caratteri del VIC-20 e mostra quali numeri occorre inserire (POKE) nella memoria dello schermo (locazioni da 7680 a 8185) per ottenere un carattere desiderato. Inoltre mostra quale carattere corrisponde ad un numero rilevato (PEEK) dallo schermo.

Le due serie di caratteri sono disponibili ma solo una serie alla volta. Ciò significa che non è possibile visualizzare simultaneamente entrambe le serie di caratteri. Si passa da una serie all'altra abbassando i tasti SHIFT e COMMODORE simultaneamente, ciò ha l'effetto di cambiare il bit 2 nella locazione di memoria 36869, il che significa a sua volta che l'istruzione POKE 36869, 240 predisporrà la serie di caratteri in maiuscolo e POKE 36869, 242 predisporrà la serie di caratteri in minuscolo.

Se si vuole eseguire qualche serio effetto di animazione, si troverà che è più facile controllare gli oggetti sullo schermo inserendoli (POKE) nella memoria dello schermo (cancellandoli mediante un inserimento di 32 che è il codice per uno spazio vuoto nella stessa locazione di memoria) che non stampandoli (PRINT) sullo schermo usando i caratteri di controllo del cursore.

Qualsiasi numero presentato sulla tabella può essere visualizzato in NEGATIVO. I caratteri in negativi non sono indicati ma il negativo di qualsiasi carattere può essere ottenuto aggiungendo 128 ai numeri indicati.

NOTA: VEDERE APPENDICE MAPPA DI MEMORIA
DELLO SCHERMO

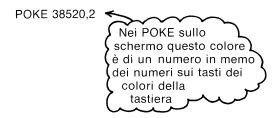
Se si vuole visualizzare un cuore nella posizione di schermo 7800, trovare il numero del carattere che si vuole visualizzare (in questo caso un cuore) in questa tabella . . . il numero per il cuore è 83 . . . quindi battere un'istruzione POKE con il numero della posizione dello schermo (7800) ed il numero del simbolo (83) come segue:

```
POKE 7800,83
```

Nell'area centrale dello schermo dovrebbe comparire un cuore bianco. Notare che esso sarà invisibile se lo schermo è bianco. Provare a cambiare la posizione battendo un numero più grande oppure battendo un simbolo diverso usando i numeri della tabella.

Se si vuole cambiare il COLORE del simbolo visualizzato, consultare la tabella che elenca i NUMERI DEI COLORI per CIASCUNA LOCAZIONE DI MEMORIA. In altre parole, per ottenere un diverso simbolo colorato in una particolare locazione, occorre un altro comando POKE.

Per esempio, per ottenere un cuore rosso, battere quanto segue:

POKE 38520,2

Ciò cambia in rosso il colore del simbolo alla posizione 7800. Se qui ci fosse un simbolo diverso, quel simbolo sarebbe ora rosso.
E' possibile visualizzare qualsiasi carattere in qualsiasi colore disponibile combinando queste due tabelle. Questi comandi POKE possono essere sommati nei programmi e risultare molto efficaci particolarmente negli effetti di animazione – ed inoltre forniscono un mezzo per osservare (PEEK) talune locazioni se si sta eseguendo programmazione sofisticata, ad esempio rimbalzi della pallina, che richiede questa informazione.

Codici dello schermo

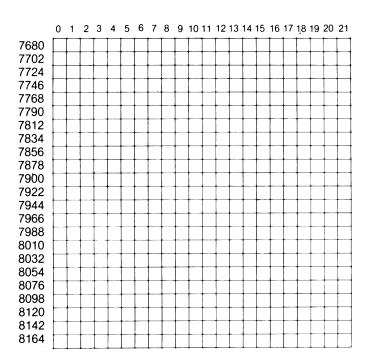
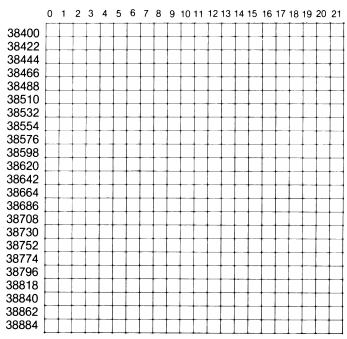
SERIE1	SERIE2	POKE	SERIE1	SERIE2	POKE	SERIE1	SERIE2	POKE
@		0	U	u	21	*		42
A	a	1	V	v	22	+		43
B	b	2	W	w	23	,		44
C	c	3	X	x	24	—		45
D	d	4	Y	y	25	.		46
E	e	5	Z	z	26	/		47
F	f	6	[		27	Ø		48
G	g	7	£		28	1		49
H	h	8	]		29	2		50
I	i	9	↑		30	3		51
J	j	10	←		31	4		52
K	k	11	SPACE		32	5		53
L	l	12	!		33	6		54
M	m	13	"		34	7		55
N	n	14	#		35	8		56
O	o	15	$		36	9		57
P	p	16	%		37	:		58
Q	q	17	&		38	;		59
R	r	18	'		39	<		60
S	s	19	(		40	=		61
T	t	20	)		41	>		62

SERIE1	SERIE2	POKE
?		63
		64
	A	65
	B	66
	C	67
	D	68
	E	69
	F	70
	G	71
	H	72
	I	73
	J	74
	K	75
	L	76
	M	77
	N	78
	O	79
	P	80
	Q	81
	R	82
	S	83
	T	84
	U	85
	V	86
	W	87
	X	88
	Y	89
	Z	90
		91
		92
		93
π		94
		95
SPACE		96
		97
		98
		99
		100
		101
		102
		103
		104
		105
		106
		107
		108
		109
		110
		111
		112
		113
		114
		115
		116
		117
		118
		119
		120
		121
	✓	122
		123
		124
		125
		126
		127

Appendice I:
Mappe di memoria dello schermo

Usare questa Appendice per trovare la locazione di memoria di qualsiasi posizione dello schermo. Basta trovare la posizione nella griglia e sommare i numeri sulla fila e sulla colonna. Ad esempio, se si vuole inserire (POKE) il carattere grafico «pallina» al centro dello schermo, basta sommare i numeri al margine della riga 11 e della colonna 11 (7900 + 10) per un totale di 7910. Se si inserisce (POKE) il codice per una pallina (81, vedere Appendice H) nella locazione 7910 battendo POKE 7910,81 sullo schermo compare una pallina bianca. Per cambiare il colore della pallina (o di altri caratteri) trovare la corrispondente posizione sulla mappa di memoria dei codici dei colori, sommare i numeri di fila e di colonna (38620 + 10, ossia 36830) per il codice di colore e battere una seconda istruzione POKE. Per esempio, se si inserisce un codice colore in questa locazione, POKE 38630,3, la pallina cambierà colore diventando blu-verde. Notare che durante l'inserimento (POKE) i numeri dei colori dei caratteri sono minori di uno dei numeri dei tasti dei colori come indicato qui di seguito.

Elenco abbreviato dei codici dei colori:

Codice	**Colore**
0	Nero
1	Bianco
2	Rosso
3	Blu-verde
4	Porpora
5	Verde
6	Blu
7	Giallo

Pagina 1: Codici dei caratteri dello schermo

Pagina 2: Mappa di memoria dei codici dei colori

Appendice J: Codici ASCII e CHR$

Questa appendice mostra quali caratteri compariranno se si esegue PRINT CHR$(X) per tutti i possibili valori di X. Essa mostrerà anche i valori ottenuti battendo PRINT ASC(«X») dove X è qualsiasi carattere che è possibile battere. Ciò è utile per valutare il carattere ricevuto in un'istruzione GET, convertendo maiuscole/minuscole e stampando comandi basati su caratteri (tipo il passaggio al maiuscolo/minuscolo) che potrebbero non essere racchiusi tra virgolette.

PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
	0		16	SPACE	32	Ø	48
	1	CRSR ↓	17	!	33	1	49
	2	RVS ON	18	"	34	2	50
	3	CLR HOME	19	#	35	3	51
	4	INST DEL	20	$	36	4	52
WHT	5		21	%	37	5	53
	6		22	&	38	6	54
	7		23	.	39	7	55
	8		24	(	40	8	56
	9		25	)	41	9	57
	10		26	*	42	:	58
	11		27	+	43	;	59
	12	RED	28	,	44	<	60
RETURN	13	CRSR ⇒	29	—	45	=	61
Passage en minuscules	14	GRN	30	.	46	>	62
	15	BLU	31	/	47	?	63

PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
@	64	U	85		106		127
A	65	V	86		107		128
B	66	W	87		108		129
C	67	X	88		109		130
D	68	Y	89		110		131
E	69	Z	90		111		132
F	70	[	91		112	f1	133
G	71	£	92	●	113	f3	134
H	72	]	93		114	f5	135
I	73	↑	94	♥	115	f7	136
J	74	←	95		111	f2	137
K	75		96		117	f4	138
I	76	♠	97		118	f6	139
M	77		98	○	119	f8	140
N	78		99	♣	120	SHIFT RETURN	141
O	79		100		121	Passage en majuscules	142
P	80		101	♦	122		143
Q	81		102		123	BLK	144
R	82		103		124	↑ CRSR	145
S	83		104		125	RVS OFF	146
T	84		105	π	126	CLR HOME	147

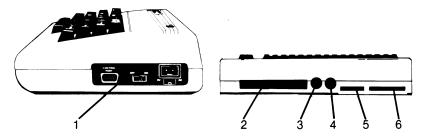
PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
INST DEL	148	CYN	159		170		181
	149	SPACE	160		171		182
	150		161		172		183
	151		162		173		184
	152		163		174		185
	153		164		175		186
	154		165		176		187
	155		166		177		188
PUR	156		167		178		189
⇐ CRSR	157		168		179		190
YEL	158		169		180		191

Appendice K:
Derivazione di funzioni matematiche

Le funzioni che non sono integrate nel BASIC VIC possono essere calcolate come segue:

FUNZIONE	EQUIVALENTE VIC BASIC
SECANTE	SEC(X) = 1/COS(X)
COSECANTE	CSC(X) = 1/SIN(X)
COTANGENTE	COT(X) = 1/TAN(X)
SENO INVERSO	ARCSIN(X) = ATN(X/SQR(− X*X + 1))
COSENO INVERSO	ARCCOS(X) = − ATN(X/SQR (− X*X + 1)) + π/2
SECANTE INVERSA	ARCSEC(X) = ATN(X/SQR(X*X − 1))
COSECANTE INVERSA	ARCCSC(X) = ATN(X/SQR(X*X − 1)) + (SGN(X) − 1)* π/2
COTANGENTE INVERSA	ARCOT(X) = ATN(X) + π/2
SENO IPERBOLICO	SINE(X) = (EXP(X) − EXP(− X))/2
COSENO IPERBOLICO	COSH(X) = (EXP(X) + EXP(− X))/2
TANGENTE IPERBOLICA	TANH(X) = EXP(− X)/EXP(X) + EXP (− X))*2 + 1
SECANTE IPERBOLICA	SECH(X) = 2/(EXP(X) + EXP(− X)
COSECANTE IPERBOLICA	CSCH(X) = 2/(EXP(X) − EXP(− X))
COTANGENTE IPERBOLICA	COTH(X) = EXP(− X)/(EXP(X) − EXP(− X)*2 + 1
SENO IPERBOLICO INVERSO	ARCSINH(X) = LOG(X + SQR(X*X + 1)
COSENO IPERBOLICO INVERSO	ARCCOSH(X) = LOG(X + SQR(X*X − 1)
TANGENTE IPERBOLICA INVERSA	ARCTANH(X) = LOG((1 + X)/(1 − X))/2
SECANTE IPERBOLICA INVERSA	ARCSECH(X) = LOG((SQR (− X*X + 1) + 1)/X)
COSECANTE IPERBOLICA INVERSA	ARCCSCH(X) = LOG((SGN(X)*SQR (X*X + 1)/X
COTANGENTE IPERBOLICA INVERSA	ARCCOTH(X) = LOG((X + 1)/(X − 1))/2

Appendice L:
Configurazione dei piedini per i dispositivi di input/output

Ecco un'immagine dei connettori di I/O sul VIC:

1. I/O Giochi
2. Espansione di memoria
3. Audio e Video
4. I/O seriale (disco)
5. Cassetta
6. Uscita per l'utente (modem)

---

1. I/O Giochi

N. PIEDINO	TIPO	NOTA
1	JOYØ	
2	JOY1	
3	JOY2	
4	JOY3	
5	POT Y	
6	LIGHT PEN	
7	+5V	MAX. 1ØØmA
8	GND	
9	POT X	

2. Espansione di memoria

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22

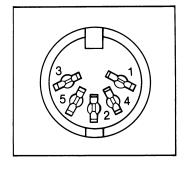
A B C D E F H J K L M N P R S T U V W X Y Z

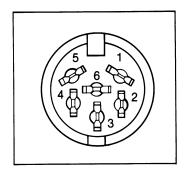
N. PIEDINO	TIPO
1	GND
2	CDØ
3	CD1
4	CD2
5	CD3
6	CD4
7	CD5
8	CD6
9	CD7
1Ø	$\overline{\text{BLK1}}$
11	$\overline{\text{BLK2}}$

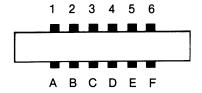
N. PIEDINO	TIPO
12	$\overline{\text{BLK3}}$
13	$\overline{\text{BLK5}}$
14	$\overline{\text{RAM1}}$
15	$\overline{\text{RAM2}}$
16	$\overline{\text{RAM3}}$
17	VR/W
18	CR/W
19	$\overline{\text{IRQ}}$
2Ø	NC
21	+5V
22	GND

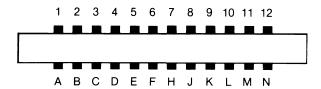
N. PIEDINO	TIPO
A	GND
B	CAØ
C	CA1
D	CA2
E	CA3
F	CA4
H	CA5
J	CA6
K	CA7
L	CA8
M	CA9

N. PIEDINO	TIPO
N	CA1Ø
P	CA11
R	CA12
S	CA13
T	I/Ø2
U	I/Ø3
V	SØ2
W	$\overline{\text{NMI}}$
X	$\overline{\text{RESET}}$
Y	NC
Z	GND

3. Audio/Video

N. PIEDINO	TIPO	NOTA
1	+6V	1ØmA MAX
2	GND	
3	AUDIO	
4	VIDEO LOW	
5	VIDEO HIGH	

4. I/O Seriale

N. PIEDINO	TIPO
1	SERIAL SRQ IN
2	GND
3	SERIAL ATN IN/OUT
4	SERIAL CLK IN/OUT
5	SERIAL DATA IN/OUT
6	NC

5. Cassetta

N. PIEDINO	TIPO
A-1	GND
B-2	+5V
C-3	CASSETTE MOTOR
D-4	CASSETTE READ
E-5	CASSETTE WRITE
F-6	CASSETTE SWITCH

6. I/O Utente

N. PIEDINO	TIPO	NOTA	N. PIEDINO	TIPO	NOTA
1	GND		A	GND	
2	+5V	1ØØmA MAX.	B	CB1	
3	$\overline{\text{RESET}}$		C	PBØ	
4	JOYØ		D	PB1	
5	JOY1		E	PB2	
6	JOY2		F	PB3	
7	LIGHT PEN		H	PB4	
8	CASSETTE SWITCH		J	PB5	
9	SERIAL ATN IN		K	PB6	
1Ø	+9V	1ØØmA MAX.	L	PB7	
11	GND		M	CB2	
12	GND		N	GND	

Appendice M: i programmi VIC da provare

```
1 REM TANK-V-UFO BY D.LATER
5 OF=30720
6 VA=9*16↑3+14:VN=9*16↑3+13
8 POKEVA+1,25
9 PRINT"{CLR}{CRSR DOWN}..." ;
20 PRINT"{HOME}** TANK VERSUS UFO **"
21 PRINT"Z-LEFT,C-RIGHT,B-FIRE"
100 PRINT"{HOME}{CRSR DOWN}..."
101 FORA=38400TO38400+22*23   :POKEA,0:NEXT:A=0
105 PRINT"{CTRL-2}    |"
110 PRINT" _{CTRL-3}  {CTRL-1}_"
115 PRINT" |OOOO| "
120 PRINT"{CRSR UP}{CRSR LEFT}";
125 IFPEEK(197)=34THENGOSUB300:A=A+1:GOTO220
130 IFPEEK(197)=33THENGOSUB300:A=A-1:GOTO220
135 IFPEEK(197)=35ANDB=0THENB=1:C=8079+A:L=1:P=15:POKEVN,158:POKEC,160:POKEC+OF
4:GOTO170
136 IFL=0THEN140
137 POKEVA,P:P=P-1:IFP=-1THENL=0:POKEVN,0
140 IFB=0THEN170
141 FORH=1TO2
145 POKEC,32:C=C-22
150 IFPEEK(C)=60ORPEEK(C)=42ORPEEK(C)=62    THEN600                 :
155 IFC<7746THENB=0:GOTO170          :
160 POKEC,46
161 NEXT
```

Legenda per i simboli speciali:

```
"" IS CLR
"" IS HOME
"" IS CRSR DOWN
"" IS CRSR LEFT
"" IS CRSR UP
"" IS CTRL-2
"" IS CTRL-3
"" IS CTRL-5
"" IS CTRL-1
```

```
170 IFD=0THEN800
172 IFD=0THEN180
174 POKEE,32:POKEE-1,32:POKEE-2,32:K=K+1
176 IFE=ITHEND=0:GOTO180
178 E=E+J: POKEE,62:POKEE-1,42:POKEE-2,60
179 IFJ=1THEN182
180 IFINT((8186-E)/22)=22-K-AANDF=0THEN F=1:G=E+21:M=21:GOTO183       :
181 GOTO183
182 IFINT((8098-E)/22)=A-KANDF=0THENF=1:G=E+23:M=23
183 IFF=0THEN125
184 POKEG,32:G=G+M
186 IFPEEK(G)<>32THEN700
187 IFG>7680+22*21THENF=0:GOTO500
189 POKEG,81:GOTO125
220 IFA<0THENA=0
221 IFA>15THENA=15
222 PRINTTAB(A)"    |"
225 PRINTTAB(A)" _▣  ■_"
230 PRINTTAB(A)" |oooo| "
235 PRINT"▯▯▯";:GOTO135
300 PRINTTAB(A)"       "
305 PRINTTAB(A)"       "
310 PRINTTAB(A)"        "
315 PRINT"▯▯▯";:RETURN
400 PRINTPEEK(197):GOTO400
500 POKEG,66:POKEG+1,78:POKEG-1,77:POKEG-20,46:POKEG-21,46:POKEG-22,46
510 POKEG-23,46:POKEG-24,46
520 FORAA=1TO100:NEXT
530 POKEG,32:POKEG+1,32:POKEG-1,32:POKEG-20,32:POKEG-21,32:POKEG-22,32
535 POKEG-23,32:POKEG-24,32
```

```
590 GOTO125
600 POKEC,160:POKEC+1,160:POKEC-1,160:POKEC+22,160:POKEC-22,160
601 L=0
610 POKEVN,128+100
611 FORGG=15TO0STEP-1:POKEVA,GG:FORGH=1TO70:NEXT:NEXT
615 B=0:D=0
616 POKEC,32 :POKEC+1, 32:POKEC-1, 32:POKEC+2, 32:POKEC-2,32:POKEC+3,32:POKEC-3
,32
617 POKEC-22,32:POKEC+22,32
640 E=E+22+J:POKEVA,15:POKEVN,0:
645 POKEE,62:POKEE-1,42:POKEE-2,60
646 FORO=248TO253:POKEVN-1,O:NEXT:FORO=253TO248STEP-1:POKEVN-1,O:NEXT
647 POKEE,32:POKEE-1,32:POKEE-2,32
650 IFE<7680+20*22THEN640
651 E=E+J
652 POKEE+22,62:POKEE+21,42:POKEE+20,60: POKEE+OF,4:POKEE+OF-1,4:POKEE+OF-2,4
653 POKEE+22+OF,0:POKEE+21+OF,0:POKEE+20+OF,32
654 POKEVN-1,0:POKEVN,128:FORQ=1TO20:POKEVA,15-INT(Q/1.33)
655 POKEE,223:POKEE-1,223:POKEE-2,223:FORO=1TO80:NEXT
656 POKEE,233:POKEE-1,233:POKEE-2,233:FORO=1TO80:NEXT
657 NEXT:POKEE,32:POKEE-1,32:POKEE-2,32:POKEE+22,32:POKEE+21,32:POKEE+20,32 :
658 PRINT"                    "
659 DU=DU+1:PRINT"UFOS"DU"TANKS"DT:PRINT""
660 GOTO125
700 POKEVN,128:L=0
701 A=A+1: FORKL=1TO200:POKEVA,15-INT(KL/13):
704 PRINTTAB(A)""
705 PRINTTAB(A)"      "
715 PRINT"";
720 PRINTTAB(A)""
```

```
725 PRINTTAB(A)"      "
735 PRINT"";
740 NEXT
745 PRINTTAB(A)"     "
750 PRINTTAB(A)"     "
751 PRINTTAB(A-1)"       ";
752 PRINT"";
756 PRINT"            ";
752 PRINT"";
756 PRINT"                     "
757 DT=DT+1:PRINT"UFOS"DU"TANKS"DT:PRINT""
760 F=0:  A=0:GOTO105
800 D=1:E=7702+INT(RND(1)*14)*22+88  :I=E-20:K=0:J=-1:IFRND(1)>.5THENE=E-21:I=E+
20:J=1
805 GOTO172
```

```
0 REM""   KILLER COMET        BY  DUANE LATER
1 C=7680+22*6:DIMB(12):LL=C:POKE9*16↑3+15,8
2 PRINT"*** KILLER COMET  ***":T=0
3 PRINT"    HIT ANY KEY"
8 POKE8179,160:
9 FORA=38444TO38400+505:POKEA,1:NEXT
10 :
20 FORA=1 TO 12:B(A)=160:NEXT
21 IFW=0THEN25
22 POKE9*16↑3+14,U:U=U-2:IFU<=0THENW=0:POKE9*16↑3+13,0
25 F=1:REM ERA METEOR
```

```
 26 FORE=0TO44STEP22
 27 FORD=C+ETOC+3+E:POKED,32  :F=F+1:NEXT:NEXT
 28 C=C+1
 29 F=1:REM DRAW METEOR
 30 POKE8179,160:FORE=0TO44STEP22
 35 FORD=C+ETOC+3+E:POKED,B(F):F=F+1:NEXT:NEXT
 36 IFPEEK(8178)=160THENPRINT"MOON BASE DESTROYED !":GOTO500
 40 GETA$:IFA$<>""ANDG=0THENG=1:S=7680+15+22*21
 50 IFG=0THEN80
 55 POKES,32:S=S-22
 60 IFS<7746THENG=0:GOTO21
 70 IFPEEK(S)=160THENPOKES,32:G=0:T=T+1:W=1:POKE9*16↑3+13,128+000:U=15:GOTO80
 71 IFPEEK(S-1)=160THENG=0:POKES-1,32:T=T+1:W=1:POKE9*16↑3+13,128+000:U=15:GOTO8
0
 75 POKES,81
 80 F=1:REM CHECK MET
 81 IFT=12THENPRINT"***METEOR DESTROYED***":FORRR=1TO2500:NEXT:LL=LL+44:C=LL
:GOTO2
 82 FORE=0TO44STEP22
 84 FORD=C+ETOC+3+E:IFPEEK(D)=32THENB(F)=32
 86 F=F+1:NEXT:NEXT
 90 GOTO21
 500 POKE9*16↑3+13,128+5
 505 POKE9*16↑3+14,5:FORRR=1TO300:NEXT
 510 FORA=15TO0STEP-1
 511 POKE9*16↑3+14,A
 520 FORRR=1TO500:NEXT
 530 NEXT
 540 FORRR=1TO2000:NEXT:RUN
READY.
```

```
1 REM"████████"
2 REM"████████"   ROCKET COMMAND       BY  DUANE LATER
10 VI=9*16↑3:OF=38400-7680:PRINT"█"
11 FORA=38400+22TO38400+22*23:POKEA,0:NEXT
15 C=7680+22*20+15
20 POKEVI+15,6+128+64+32+8
30 PRINT"██*** ROCKET COMMAND ***";
35 PRINT"      HIT ANY KEY"
40 PRINT"█████████████████████████████████○○○○○○○○○○○○○○○○";
50 PRINT"█████████████████↑↑↑↑↑↑↑↑↑↑↑↑↑↑"
60 A=7680+22*22+15
70 GET A$:IFA$<>""ANDB=0THENB=1:POKEC+22,32:D=C:C=C-1:K=1:POKE9*16↑3+13,128+125
:L=16
71 IFK=1THENL=L-1:POKE9*16↑3+14,L
72 IFL=0THENK=0:POKE9*16↑3+13,0
75 IFC=8121THENPRINT"████████NO MORE ROCKETS!!!":POKE9*16↑3+14,0: FORAA=1TO3000:
NEXT:RUN
80 IFB=0THEN110
85 POKED,32:D=D-22
90 IFPEEK(D)=60ORPEEK(D)=62THENGOSUB500:GOTO70
100 IFD<7680+88THENB=0:GOTO110
105 POKED,30
110 IFH=0THEN200
115 IFH=0THEN70
120 POKEF,32:F=F+E
125 IFF=ITHENH=0:GOTO70
130 IFPEEK(F)=30THENGOSUB500:GOTO70
140 POKEF,G:GOTO70
200 H=1
205 IFINT(RND(1)*2)=1THENE=-1:F=7702+(INT(RND(1)*10)+6)*22:I=F-22:G=60:  GOTO115
```

```
210 E=1:F=7680+(INT(RND(1)*10)+6)*22:I=F+22:G=62: GOTO115
500 B=0:H=0
501 SC=SC+10:PRINT"[illegible]SCORE = "SC
502 POKEF+OF,4:POKEF+1+OF,4:POKEF-1+OF,4
503 POKEF+OF+22,4:POKEF-22+OF,4
510 POKEF,160:POKEF+1,160:POKEF-1,160:POKEF+22,160:POKEF-22,160
521 POKE9*16↑3+13,128+35
522 FORY=16TO0STEP-1
523 POKE9*16↑3+14,Y
524 FORP=1TO80:NEXT:NEXT
530 POKEF,32:POKEF+1,32:POKEF-1,32:POKEF+22,32:POKEF-22,32
531 POKEF+OF,0:POKEF+1+OF,0:POKEF-1+OF,0
532 POKEF+OF+22,0:POKEF-22+OF,0
533 POKE9*16↑3+13,0
540 FORGH=FTOF+22*16STEP22
544 II=PEEK(GH):POKEGH,G:FOROO=1TO60:NEXT
546 POKEGH,II:NEXT
800 RETURN
999 GOTO70
1000 POKE9*16↑3+13,128+125
1001 FORY=16TO0STEP-1
1005 POKE9*16↑3+14,Y
1010 NEXT:POKE9*16↑3+13,0
1020 RETURN
READY.
```

Appendice N: Messaggi di errore

Questa Appendice contiene un elenco completo dei messaggi di errore generati dal VIC, con una descrizione delle relative cause.

BAD DATA (DATI ERRATI). La stringa di dati sta ricevendo da un file aperto, ma il programma stava aspettando dati numerici.

BAD SUBSCRIPT (INDICE ERRATO). Il programma stava cercando di fare riferimento ad un elemento di una matrice il cui numero è al di fuori del campo specificato nell'istruzione DIM.

CAN'T CONTINUE (NON POSSO CONTINUARE). Il comando CONT non funziona in quando il programma non è mai stato eseguito (RUN), c'è stato un errore o è stata corretta una riga.

DEVICE NOT PRESENT (DISPOSITIVO NON PRESENTE). Il dispositivo di I/O richiesto non è disponibile per un'istruzione OPEN, CLOSE, CMD, PRINT#, INPUT# o GET#.

DIVISION BY ZERO (DIVISIONE PER ZERO). La divisione per zero è un non senso matematico e non è ammessa.

EXTRA IGNORED (IGNORATI GLI EXTRA). Sono stati battuti troppi elementi di dati in risposta ad un'istruzione INPUT. Sono stati accettati solo i primi elementi.

FILE NOT FOUND (FILE NON TROVATO). Se si sta cercando un file su nastro, è stato trovato il marcatore di fine nastro. Se si cerca sul disco, non esiste un file di quel nome.

FILE NOT OPEN (FILE NON APERTO). Il file specificato in un'istruzione CLOSE, CMD, PRINT#, INPUT# o GET#, deve essere per prima cosa aperto (OPEN).

FILE OPEN (FILE APERTO). E' stato compiuto un tentativo di aprire un file usando il numero di un file già aperto.

FORMULA TOO COMPLEX (FORMULA TROPPO COMPLESSA). L'espressione stringa che viene valutata deve essere divisa in almeno due parti perchè il sistema la possa elaborare.

ILLEGAL DIRECT (DIRETTO ILLECITO). L'istruzione INPUT può essere usata soltanto nell'ambito di un programma e non nel modo diretto.

ILLEGAL QUANTITY (QUANTITA' ILLECITA). Un numero usato come argomento di una funzione o istruzione è fuori dal campo ammesso.

LOAD (CARICAMENTO). C'è un problema con il programma sul nastro.

NEXT WITHOUT FOR (NEXT SENZA FOR). Ciò è provocato dalla nidificazione scorretta di iterazioni o dalla presenza di un nome variabile in un'istruzione NEXT che non corrisponde ad una variabile in un'istruzione FOR.

NOT INPUT FILE (NON FILE DI INPUT). E' stato compiuto un tentativo di INPUT o GET dati dal file che era specificato soltanto per l'output.

NOT OUTPUT FILE (NON FILE DI OUTPUT). E' stato compiuto un tentativo di stampare (PRINT) i dati su un file che era specificato per la sola immissione.

OUT OF DATA (MANCANO DATI). E' stata eseguita un'istruzione READ ma non ci sono dati da leggere in un'istruzione DATA.

OUT OF MEMORY (MANCA MEMORIA). Non c'è altra RAM disponibile per il programma o per le variabili. Ciò può anche verificarsi quando sono state nidificate troppe iterazioni FOR oppure quando ci sono troppi GOSUB in atto.

OVERFLOW (SUPERO DI CAPACITA'). Il risultato di un calcolo è maggiore del numero massimo ammesso che è 1.70141884E + 38.

REDIM'D ARRAY (MATRICE RIDIMENSIONATA). Una matrice può essere dimensionata (DIM) soltanto una volta. Se viene usata una variabile matrice prima che la matrice sia dimensionata, viene eseguita un'operazione di dimensionamento (DIM) automatica su quella matrice definendo il numero di elementi a dieci. Qualsiasi successiva DIM provoca questo errore.

REDO FROM START (RIFARE DALL'INIZIO). Sono stati battuti dati alfanumerici in un'istruzione INPUT quando si aspettavano dati numerici. Basta ribattere l'entrata corretta ed il programma continuerà da solo.

RETURN WITHOUT GOSUB (RETURN SENZA GOSUB). E' stata incontrata un'istruzione RETURN e non è stato emesso alcun comando GOSUB.

STRING TOO LONG (STRINGA TROPPO LUNGA). Una stringa può contenere fino a 255 caratteri.

SYNTAX (SINTASSI). Un'istruzione non è riconosciuta dal VIC. Ci sono parentesi in più o ne mancano, ci sono parole chiave scritte erroneamente, ecc.

TYPE MISMATCH (MANCATA CORRISPONDENZA DI TIPO). Questo errore si verifica quando viene usato un numero in luogo di una stringa o viceversa.

UNDEF'D FUNCTION (FUNZIONE INDEFINITA). Si è fatto riferimento ad una funzione definita dall'utente che però non è stata mai definita usando l'istruzione DEF FN.

UNDEF'D STATEMENT (ISTRUZIONE INDEFINITA). E' stato compiuto un tentativo di GOTO o GOSUB o RUN su un numero di riga che non esiste.

VERIFY (VERIFICA). Il programma sul nastro o su disco non corrisponde al programma correntemente in memoria.

Indice alfabetico

G

I

M

N

O

P

R

S

T

U

V

Il nuovo VIC computer

è stato realizzato per essere un valido amico per l'utente . . . amico nel prezzo, amico nella forma, amico da usare ed espandere.

Con il VIC, la COMMODORE mette a disposizione un sistema che permette a chiunque di apprendere facilmente e velocemente i concetti fondamentali del Computer . . . con la predisposizione di espandere il sistema, quando le esigenze e le conoscenze dell'utente diventano più sofisticate.

I possessori del VIC che desiderano imparare di più circa la programmazione, dovrebbero rivolgersi al loro rivenditore Commodore per avere informazioni riguardanti la seguente letteratura e materiali di riferimento:

- **SERIE APPRENDIMENTO VIC . . .**
Una libreria di libri, nastri, cartucce che aiutano ad entrare nel mondo dei Computer.
Il primo volume della serie «Apprendimento» è chiamato «Introduzione al Basic Parte I» a questo ferà seguito la seconda parte, che darà informazioni sull'animazione, suono, musica ed altro.

- **VIC PROGRAMMER'S REFERENCE GUIDE . . .**
Un altro volume con importanti informazioni per nuovi o esperti programmatori è il VIC Programmer's reference guide.

- **PROGRAMMI PER IL VIC SU NASTRI, CARTUCCE, E DISCHI . . .**
Una libreria che si arricchisce di giorno in giorno, con nuovi programmi di educazione scolastica, utilità domestica, che aiuterranno a risolvere i vari problemi e aumenteranno la conoscenza; infine nuovi giochi per trascorrere ore liete.
Questi programmi facili da usare non richiedono esperienza nella elaborazione dati.